Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática
2010

Júlio César Ferreira

CARACTERIZAÇÃO ESTATÍSTICA DE ATAQUES DISTRIBUIDOS EM REDES LOCAIS

Universidade de Aveiro
2010
Departamento de Electrónica, Telecomunicações e Informática

Júlio César Ferreira

CARACTERIZAÇÃO ESTATÍSTICA DE ATAQUES DISTRIBUIDOS EM REDES LOCAIS

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações, realizada sob a orientação científica do Doutor António Nogueira, Professor Auxiliar e do Doutor Paulo Salvador, Professor Auxiliar, ambos do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

Dedico este trabalho aos meus pais e amigos.

o júri

presidente

Professora Doutora Ana Maria Perfeito Tomé
Professora Associada da Universidade de Aveiro

Professor Doutor Rui Jorge Morais Tomaz Valadas
Professor Catedrático da Universidade Técnica de Lisboa

Professor Doutor António Manuel Duarte Nogueira
Professor Auxiliar da Universidade de Aveiro

Professor Doutor Paulo Jorge Salvador Serra Ferreira
Professor Auxiliar da Universidade de Aveiro

agradecimentos

Aos meus pais, César Augusto Ferreira e Maria Cesaltina dos Anjos Ferreira, pelos seus ensinamentos e por nunca terem deixado de confiar e acreditar no meu trabalho ao longo dos anos, mesmo nos piores momentos.

A todos os que tenho orgulho de chamar de amigos e que nunca me permitiram duvidar das minhas capacidades e sempre estiveram presentes neste longo caminho, em especial nesta ultima etapa à Susana Barroso que tanto me apoiou.

Por incrível que pareça, aos meus animais de estimação que nunca recusaram ouvir os meus lamentos ou partilhar alegrias e tristezas.

A todos os professores que ao longo de vários anos me transmitiram os seus conhecimentos, especialmente ao meu orientador Prof. Doutor António Manuel Duarte Nogueira e ao meu co-orientador Paulo Jorge Salvador Serra Ferreira pelo despertar de interessa neste área e por toda a ajuda e conhecimento fundamentais para a concretização desta dissertação.

palavras-chave IDS, Correlação, Rede, Port-scan, Ataques, Estatística, Detecção

resumo

O enorme crescimento da Internet, do número de serviços e da heterogeneidade das tecnologias de rede têm sido acompanhados por um cada vez maior número de ameaças à segurança das redes e dos seus utilizadores.

A maioria dos ataques começa com a aquisição de informações acerca da rede, das suas máquinas e utilizadores.

A heterogeneidade e complexidade das redes actuais tornam impossível uma gestão e manutenção exclusivamente humana. Torna-se por isso necessária a existência de ferramentas capazes de auxiliar nas tarefas de gestão e manutenção das redes, que possam de forma independente fazer a aquisição, correlação e filtragem de informação dispersa nos vários equipamentos de rede, de modo a que ao administrador de rede apenas seja passada a informação mais relevante. Estas informações devem ser apresentadas de forma estruturada, permitindo uma rápida e fácil detecção de ataques e intrusões. Torna-se então vital o estudo detalhado de técnicas de detecção de intrusão.

Nesta dissertação é apresentada uma caracterização estatística de diversos ataques à segurança em múltiplos cenários de rede. Concretamente, para cada tipo de ataque são inferidas as distribuições de probabilidade que melhor ajustam as principais características do tráfego gerado pelas interacções inerentes ao ataque.

keywords IDS, Correlation, Network, Port-scan, Attacks, Statistics, Detection.

abstract

The tremendous growth of the Internet, the high number of services and the heterogeneity of network technologies have been accompanied by an increasing number of security threats to networks and their users. The vast majority of network attacks start by the acquisition of information about the network, their machines and users. The heterogeneity and complexity of current networks make an exclusively human management and maintenance virtually impossible. It is necessary to have tools that can aid in network management tasks, that can independently make the acquisition of information spread amongst various network equipments, correlate and filter them, so that the network administrator can only see the most relevant information. This information should be presented in a structured way, allowing an easy and rapid intrusion detection. Therefore, having a detailed description of each intrusion technique is vital.

This dissertation presents a statistical characterization of several security attacks in multiple network scenarios. Specifically, for each type of attack we want to infer the probability distribution that best fits the main characteristics of the traffic generated by the attack interactions.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE ACRÓNIMOS E SIGLAS

ACK – Acknowledge

CERT / CC – Computer Emergency Response Team/Coordination Center

DDoS – Distributed Denial of Service

DNS – Domain Name System

DoS – Denial of Service

FDP – Função Densidade De Probabilidade

FTP – File Transfer Protocol

GNS3 – Graphical Network Simulator

HIDS – Host Based Intrusion Detection System

HTTP – Hypertext Transfer Protocol

IANA – Internet Assigned Numbers Authority

ICMP – Internet Control Message Protocol

IDS – Intrusion Detection System

IOS – Internetwork Operating System

IP – Internet Protocol

ISO – International Organization For Standardization

MATLAB – Matrix Laboratory

NIDS – Network Based Intrusion Detection System

NIST – National Institute Of Standards And Technology

P2P – Peer To Peer

POP3 – Post Office Protocol Version 3

R2L – REMOTE TO LOCAL

RST - Reset

SMTP – Simple Mail Transfer Protocol

SNMP – Simple Network Management Protocol

SYN – Synchronize

TCP – Transmission Control Protocol

U2R – User To Root

UDP – User Datagram Protocol

1 – Introdução

A necessidade de segurança não é algo recente, ela existe desde a introdução do primeiro computador.

O enorme crescimento da Internet, do número de serviços e da heterogeneidade das tecnologias de rede têm sido acompanhados por um cada vez maior número de ameaças à segurança das redes e dos seus utilizadores.

O acesso à Internet tem crescido nos últimos anos e as organizações e empresas têm, de forma crescente, colocado informações críticas online. Tal fez com que as actividades dos cyber criminosos aumentasse e, virtualmente, todas as organizações enfrentam crescentes ameaças aos seus recursos da rede e aos serviços que prestam [1].

O tamanho que a rede adquiriu nos dias de hoje torna impossível uma gestão e manutenção exclusivamente humana. Torna-se então necessária a existência de ferramentas auxiliares, que de forma autónoma façam a recolha da informação dispersa pelos diversos equipamentos de rede, a correlacionem e filtrem, de modo a que ao administrador apenas seja apresentada a informação relevante de forma estruturada.

1.1– Modelos de gestão

A International Organization for Standardization (ISO) definiu cinco áreas conceptuais na gestão de redes. Isolando os problemas de gestão em áreas distintas, este modelo permite conceptualizar soluções que são optimizadas para os problemas específicos de cada área funcional.

Estas áreas resumem-se à sigla FCAPS:

- Fault Manegement (gestão de falhas): detecção, isolamento e, se possível, correcção de comportamentos anormais;
- Configuration Management (gestão de configuração): monitorização e configuração para gestão de várias versões de hardware e software;
- Accounting Manegement (gestão de contabilidade): medição de parâmetros como a utilização da rede para determinar custos, de maneira a regular e minimizar problemas da rede;
- Performance Manegement (gestão de desempenho): avaliação e medição de desempenho da rede;
- Security Manegement (gestão de segurança): controlar o acesso seguro aos recursos da rede.

Este trabalho insere-se na área do Security Management

1.2 – Motivação

A motivação por detrás desta dissertação assenta na necessidade de libertar e facilitar o trabalho dos gestores de rede na detecção e tomada de decisões aquando da ocorrência de falhas de segurança na rede.

A análise do tráfego de rede recorrendo apenas ao olho humano torna-se uma tarefa demorada e penosa, sendo quase impossível uma detecção atempada e eficaz de eventuais falhas de segurança, bem como uma tomada de decisão rápida e eficiente.

Nesta dissertação pretende-se estudar técnicas de detecção de intrusões, nomeadamente *port scans* quando não é possível aceder ao tráfego da rede e recorrendo apenas a dados estatísticos do tráfego ou seja não tendo acesso ao tráfego em si. O acesso total ao tráfego, do ponto de vista de detecção de intrusões, implica grande

capacidade de armazenamento de dados, um elevado tempo de processamento, bem como acesso a tudo o que circula na rede o que pode levantar problemas de confidencialidade. Assim a opção de obter dados estatísticos do tráfego gerado que são depois passados para análise e detecção de anomalias elimina a necessidade de armazenar grandes quantidades de dados, uma vez que as estatísticas do tráfego podem ser guardadas em ficheiros de pequena dimensão e do mesmo modo estes ficheiros podem ser processados com relativa rapidez. O facto de o tráfego ser convertido em dados estatísticos ultrapassa também o problema de confidencialidade pois informações que possam ser confidencias não são passadas para análise.

A forma como são obtidas as estatísticas do tráfego será discutida no capitulo 3.

1.3 Estrutura da dissertação

Capitulo 1 – **Introdução**

Capítulo 2 – **Estado da arte** – é feita uma descrição do ambiente geral que envolve um ataque, desde a própria definição de ataque até à categorização dos atacantes. É adoptada uma taxonomia para os ataques à rede, que contempla tipos de ataques, número de ligações envolvidas, fonte do ataque, ambiente e nível de automatização, e dada de forma resumida uma descrição de cada um dos tipos citados. É ainda feita uma abordagem aos diferentes tipos de *Instrusion Detection Systems* (IDS).

Capítulo 3 – **Metodologia** – é apresentada a forma como o problema é abordado e as ferramentas utilizadas para a obtenção de dados e seu tratamento.

Capítulo 4 – **Obtenção de Dados** – são apresentados os dados obtidos através da análise e filtragem do tráfego disponível.

Capítulo 5 – **Resultados** – são dados a conhecer os resultados obtidos através dos dados recolhidos e comparações entre os mesmos resultados.

Capítulo 6 – **Conclusões** – são apresentadas as conclusões decorrentes da análise dos resultados apresentados no capítulo 5.

Capítulo 7 – **Trabalho Futuro** – são apresentadas algumas sugestões para trabalhos a realizar no futuro.

2 - Estado da arte

O conceito de segurança está definitivamente instalado no centro das atenções de qualquer administrador de redes. Análises à segurança da Internet realizadas por diversas entidades apontam num único sentido: as ameaças são cada vez maiores e acontecem com mais frequência, como se pode ver através da figura 1 criada a partir de dados obtidos pelo Computer Emergency Response Team/Coordination Center (CERT/CC) [2]. De notar que dado o amplo uso de ferramentas de ataque automatizadas, os ataques contra sistemas ligados à Internet tornaram-se tão comuns que a contagem do número de incidentes reportados fornecem pouca informação no que diz respeito à avaliação do alcance e impacto dos ataques. Por isso, essa estatística deixou de ser fornecida a partir do final de 2003. Em adição, a severidade e sofisticação dos ataques também estão a crescer (figura 2).

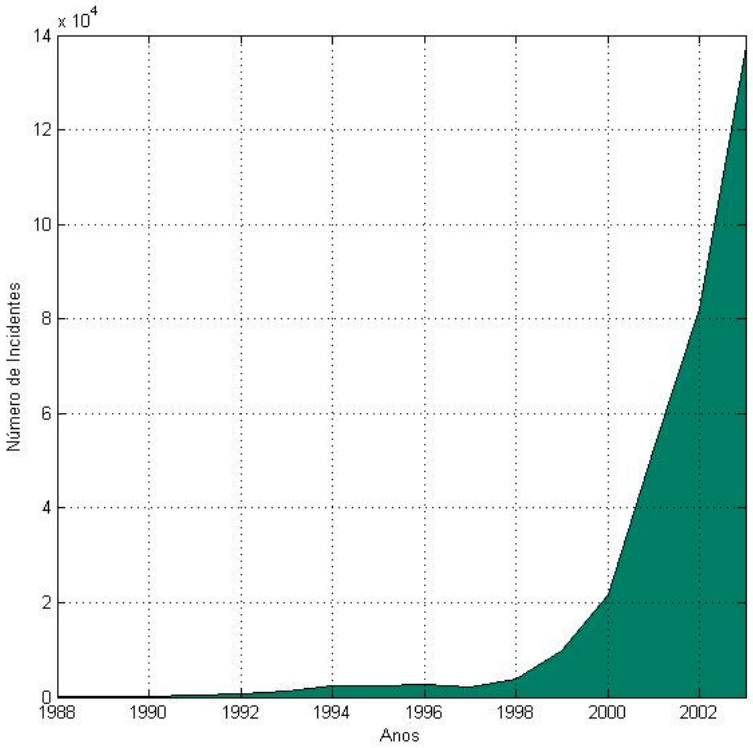

Figura 1 – Número de incidentes reportados ao CERT

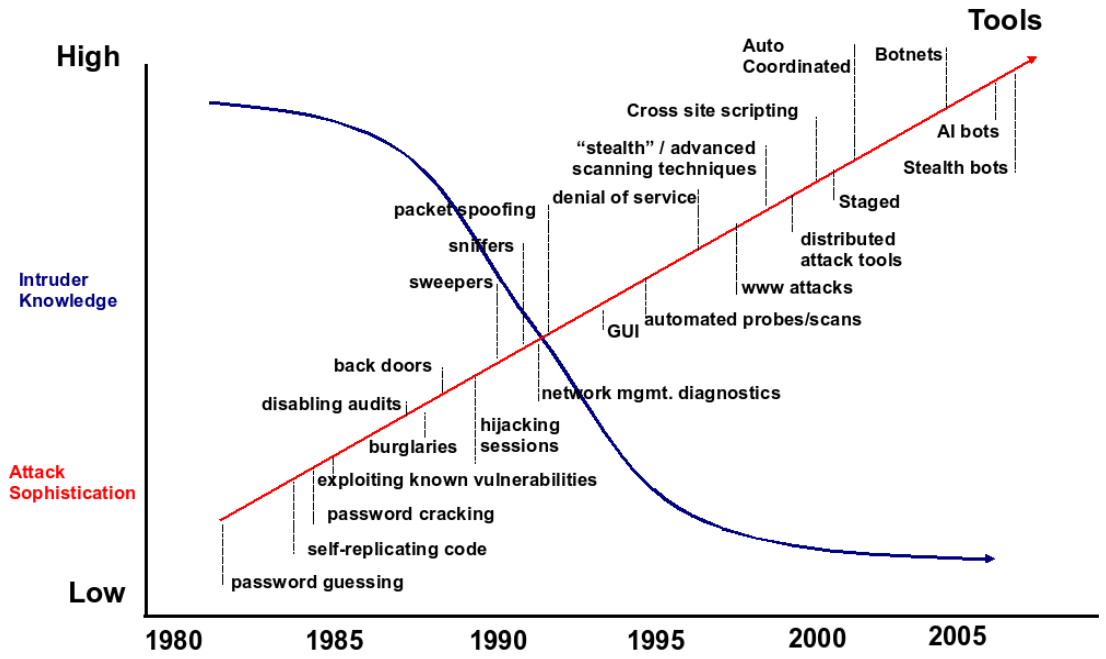

Figura 2 – Sofisticação dos ataques versus conhecimento dos atacantes.

2.1 – Conceito/definição de intrusão

O National Institute of Standards and Technology (NIST) classificou em 2001 [3] detecção de intrusão como: " *the process of monitoring the events occurring in a computer system or network and analyzing then for signs of instructions, defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network*".

O conceito de detecção de intrusão nasceu em 1980 com o artigo de James Anderson, Computer Security Threat Monitoring and Surveillance [4]. Anderson, ao introduzir o conceito de detecção de intrusão, definiu uma tentativa de intrusão ou ameaça como sendo a possibilidade de ocorrência de uma tentativa deliberada e não autorizada para:

- Ter acesso a informação;

- Manipular informação;
- Tornar um sistema não confiável ou indisponível.

Anderson, no referido artigo, apresentou também alguns conceitos relacionados com os riscos e ameaças que afectam a segurança dos sistemas de informação. Estes são descritos como se segue:

- **Ameaça**: possibilidade de ocorrência de uma tentativa deliberada e não autorizada para ter acesso a informação, manipular informação ou tornar um sistema não confiável ou indisponível;
- **Risco**: exposição acidental e imprevista de informação, ou violação da integridade das operações relacionadas com o mau funcionamento do hardware ou projecto incorrecto/incompleto de software.
- **Vulnerabilidade**: uma suspeita de falha ou uma falha conhecida no projecto do hardware ou software, ou operação de um sistema que expõe informação.
- **Ataque**: formulação específica ou execução de um plano que explora uma ameaça.
- **Intrusão**: ataque bem sucedido, habilidade de obter acesso não autorizado a arquivos e programas, ou controle de um sistema de informação.

Desde então, a comunidade científica na área da segurança informática tem desenvolvido várias definições de ataque e intrusão. Sendo vulgarmente consideradas como sinónimo, a literatura acerca de detecção de intrusões distingue ataque de intrusão. Por exemplo, um sistema pode ser atacado, sem sucesso. Neste caso, o ataque existe mas o "escudo" que protege o sistema é suficientemente eficaz para prevenir uma intrusão. Um ataque é, portanto, uma tentativa de

intrusão e uma intrusão é o resultado de um ataque que foi (pelo menos parcialmente) bem sucedido (figura 3).

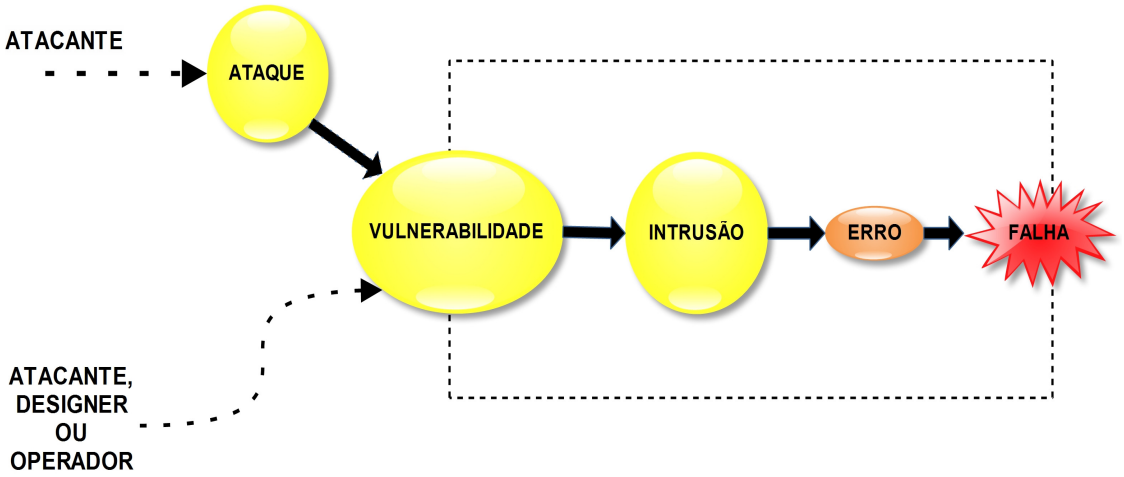

Figura 3 – Ataque VS Intrusão

(Adaptado de *Conceptual Model and Architecture of MAFTIA* [5])

2.2 – Definição dos Atacantes

Outra definição importante diz respeito às pessoas que efectuam um ataque ou intrusão. São normalmente conhecidas como executantes, atacantes, intrusos ou também como *crackers* ou *hackers*, sendo esta última denominação erradamente usada, mas de uso generalizado. O crescente uso e disponibilidade da Internet fez surgir um novo tipo de atacante conhecido como *Script Kiddies*, sendo este o atacante mais comum actualmente [6]. São normalmente utilizadores, legítimos ou não, que exploram ou tentam explorar vulnerabilidades do sistema para assim comprometer recursos, fraudar, roubar, sabotar e ter acesso a informações de forma não autorizada.

Os atacantes internos representam a ameaça mais séria, pois as suas motivações e acções podem resultar em danos desastrosos para a companhia em que estão inseridos. Os atacantes externos são

normalmente motivados pelos desafios técnicos, ganhos financeiros ou pelo reconhecimento dos seus pares.

2.3 - Taxonomia dos Ataques

Têm surgido, ao longo dos anos, várias tentativas para categorizar e classificar os diversos tipos de ataques e intrusões. Das várias tentativas surgiram diversas taxonomias centradas em diversos aspectos. Inicialmente estes aspectos focavam-se nas fraquezas dos sistemas informáticos e falhas de design nos sistemas operativos; mais tarde passaram a focar-se em aspectos como a categorização do abuso dos sistemas e categorização das pessoas que tentam acessos não autorizados, mas não existe um padrão ou norma geralmente aceite.

A taxonomia apresentada nesta dissertação será dada de uma forma generalizada e apoiada na combinação de trabalhos relativos à caracterização e taxonomia de ataques, publicados na literatura sobre a detecção de intrusões [7].

Em taxonomias já publicadas, as categorias usadas para a classificação de ataques são *normal*mente ou a causa ou o efeito da vulnerabilidade. Na taxonomia que aqui se apresenta são usadas as tradicionais causas das vulnerabilidades para especificar as seguintes categorias de ataques:

- Tipo de ataque;
- Número de ligações envolvidas no ataque;
- Fonte do ataque;
- Ambiente;
- Nível de automatização.

2.3.1 – Tipo de ataque

Trata-se do critério mais comum para a classificação dos ataques e intrusões e pode ser subdividido nos seguintes tipos:

2.3.1.1 – *Denial of Service* (DoS)

Ataques do tipo *Denial of Service* têm por objectivo incapacitar, desligar ou tornar lentas redes alvo, serviços ou processos [3].

Ataques do tipo *Denial of Service* tentam desligar/incapacitar redes, computadores ou processos; ou também negar o uso de recursos ou serviços a utilizadores, sejam eles um servidor de e-mail, servidor Web ou servidor de dados [6, 7]. São normalmente feitos, de forma intencional e maliciosa, contra sistemas ou redes específicas, quer por motivos pessoais, quer para, simplesmente, atingir organizações de alto nível.

Este tipo de ataques podem-se dividir em dois tipos: (1) ataques a sistemas operativos, que têm por alvo erros/*bugs* dos sistemas operativos; (2) ataques à rede, que exploram as limitações inerentes aos protocolos de rede e às infra-estruturas.

Um exemplo de ataque a sistemas operativos é o *Teardrop*; problemas (popular nos sistemas operativos Windows 3.1 e 95) com o código de reagrupamento dos fragmentos TCP/IP (figura 4) criaram esta abertura [8]. O protocolo IP prevê fragmentar os pacotes de grande dimensão em vários pacotes IP, possuindo cada um um número de sequência e um número de identificação comum. Na recepção dos dados, o destinatário reagrupa os pacotes graças aos valores de desfasamento (*offset*) que contêm. O princípio do ataque *Teardrop* consiste em inserir em pacotes fragmentados as informações de desfasamento erradas. Assim, aquando do

reagrupamento, existem vazios ou verificações (*overlapping*), podendo provocar uma instabilidade do sistema. Em Setembro de 2009, uma vulnerabilidade no Windows Vista foi referida como sendo um ataque *Teardrop* [9, 10].

Como exemplo de DoS à rede temos, como mais comum, ataques *SYN flooding* que tiram partido do aperto de mão em três etapas (*Three-Way Handshake*) (figura 5) do protocolo TCP para estabelecimento de ligação em que a fonte (*Source*) envia um número de sequência inicial X com o primeiro datagrama SYN, na segunda mensagem o destinatário (*Destination*) reconhece o primeiro datagrama com um ACK de número X+1 e envia o seu próprio número de sequencia Y através com um datagrama SYN, por fim a fonte reconhece o destinatário e através do envio de um datagrama ACK com número Y+1 completa o inicio da ligação. Neste ataque, o atacante cria um elevado número de meias ligações (*half open connections*) (figura 6) usando IP forjados. O atacante primeiro envia pacotes SYN com um falso endereço IP para a vítima, de forma a iniciar a ligação. A vítima regista o início de ligação na sua estrutura de dados e responde com mensagens SYN/ACK, mas nunca recebe a correspondente mensagem ACK para estabelecer a ligação, pois os endereços IP forjados pelo atacante não estão ao alcance da vítima ou são incapazes de responder. Apesar do registo de dados para ligações não concluídas ser limpo após um determinado tempo, o atacante gera um elevado número de *half open connections* sobrecarregando a estrutura de dados da vítima, levando ao bloqueio do computador.

Um dos primeiros ataques DoS, considerado também como um *Worm,* ocorreu em 1988 e ficou conhecido como Morris Worn e incapacitou a Internet de então. Um erro em parte do código criado por Robert Morris, fez com que este se replicasse de forma tão rápida

que consumia todos os recursos disponíveis e se espalhava a outras máquinas na Internet [6, 11, 12].

Uma conhecida e avançada variação dos ataques DoS, chamada *Distributed Denial of Service* (DDoS), faz uso de múltiplas máquinas para implantar o ataque. As máquinas usadas no ataque, sem o conhecimento do legítimo dono, são controladas pelo atacante e actuam como uma só, criando um imenso sistema de ataque. Incapacitar um grande sistema ou rede é impossível usando um só computador, contudo fazendo uso de um elevado número de computadores a tarefa torna-se simples [3].

O ataque ocorrido em 2000 contra a Amazon.com e à CNN.com é um bom exemplo de um ataque do tipo DDoS [13].

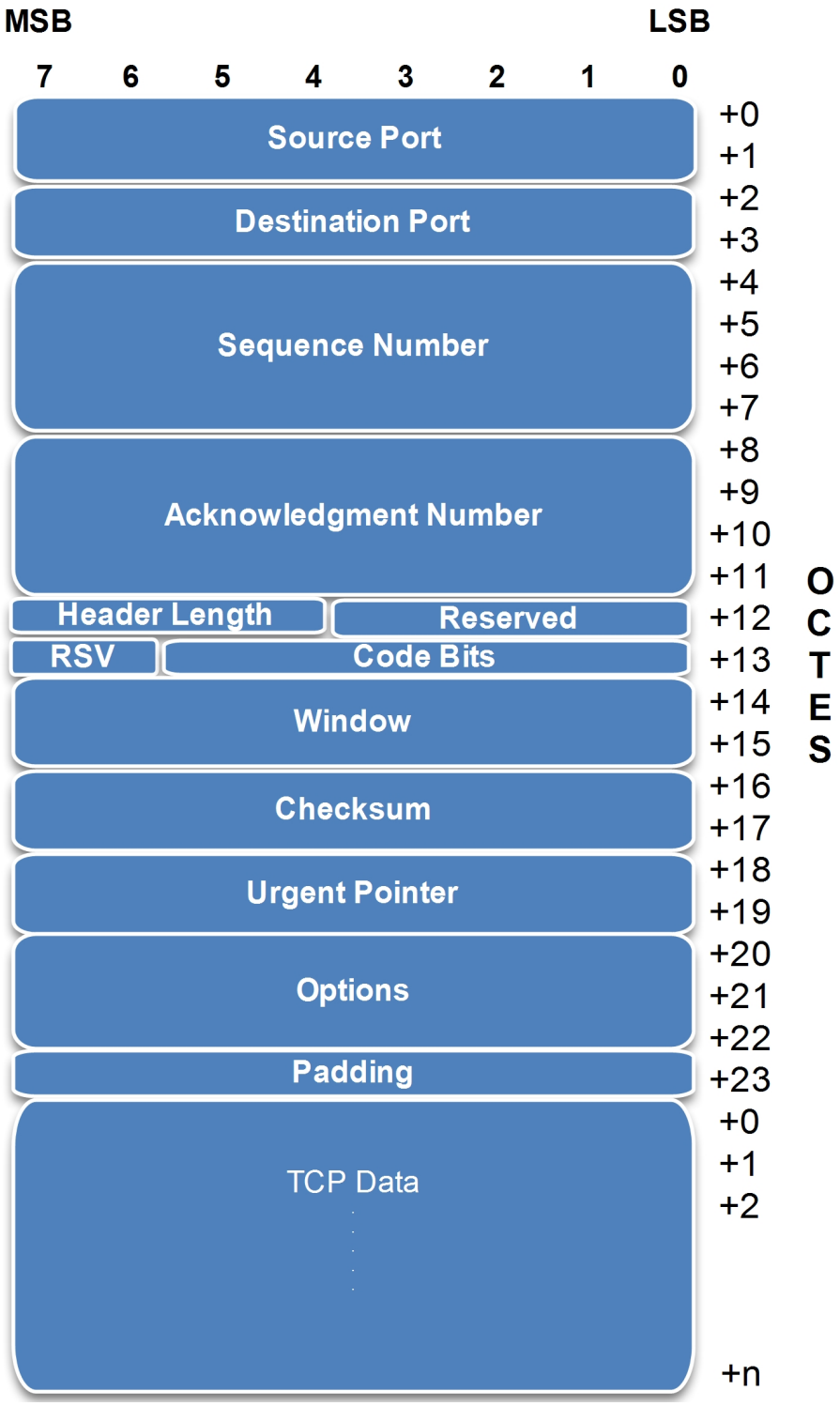

Figura 4 – Formato de um segmento TCP

(Adaptado de *A guide to the TCP/IP protocol suite* [14])

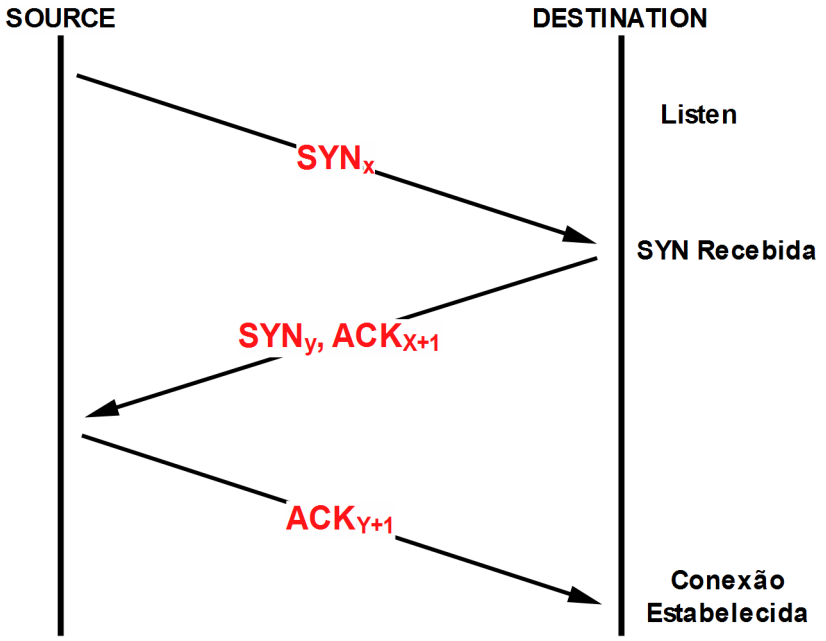

Figura 5 – *Three-Way Handshake* do Protocolo TCP

(Adaptado *de Analysis of a Denial of Service Attack on TCP* [15])

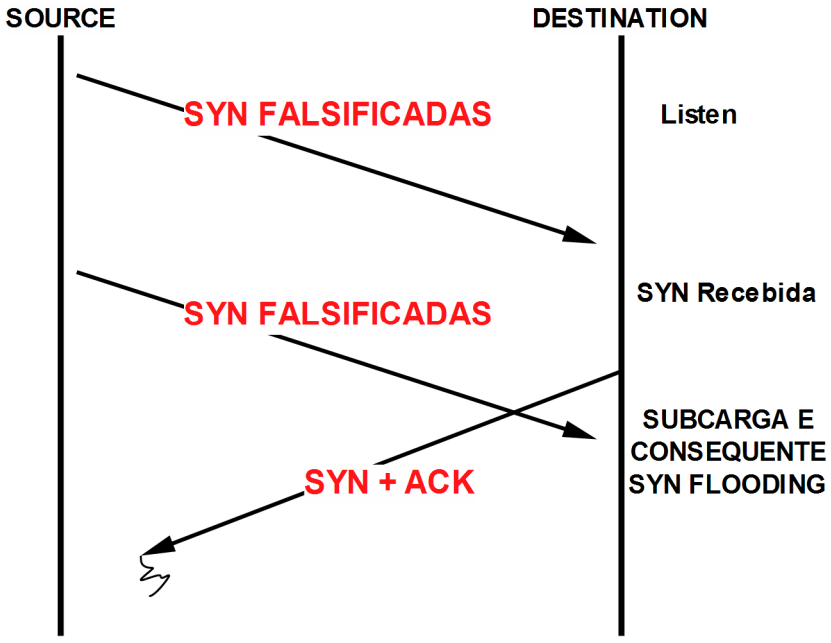

Figura 6 – Esquema de um ataque SYN Flooding

(Adaptado de *Analysis of a Denial of Service Attack on TCP* [15])

Outros ataques do tipo DoS, e funcionando de forma semelhante, são possíveis com outros protocolos (como por exemplo UDP e ICMP), onde uma mensagem leva a outra mensagem que não

é enviada como resposta ao endereço de origem da primeira mensagem [16].

2.3.1.2 – *Probing* (*surveilance, scanning*)

Este tipo de ataque é de particular importância pois faz um *scan* à rede e recolhe informação acerca desta. É provavelmente o tipo de ataque mais comum na rede, é o antecessor de outros tipos de ataque e fornece ao atacante uma lista de potenciais vulnerabilidades que depois poderá explorar.

Durante este tipo de ataque o atacante pode recolher vários tipos de informação, tais como:

- Topologia da uma rede alvo;
- Tipos de tráfego permitido através de uma *firewall*;
- *Hosts* (endereços IP) activos numa rede;
- Sistema operativo usado pelos *hosts*;
- Serviços TCP/UDP em uso.

Com este tipo de informação, um atacante pode, com precisão, identificar potenciais vítimas numa rede alvo, assim como determinar que tipo de ataque realizar para penetrar na rede.

Infelizmente para as vítimas, assim como é legal uma pessoa entrar num banco e observar o sistema de segurança deste, alguns advogados alegam que é legal fazer *scan* a um *host* ou à rede. Da perspectiva de alguém que faz um *scan*, está legalmente a 'escutar' a internet para procurar recursos de acesso público [3].

Exemplos de ataques do tipo *probing* incluem *PING Sweeps* e *Ports Scanning*.

Num *PING Sweep*, o atacante tem por objectivo ver que máquinas dentro de uma rede estão activas ou não. Existem várias formas de conseguir tal objectivo, tais como *ICMP Sweep* em que o atacante envia um pacote *ICMP ECHO request* para uma rede de máquinas (normalmente especificado como uma gama de endereços IP) e espera para ver se recebe um *ICMP ECHO reply*. Se receber este *ICMP ECHO reply* significa que o alvo está activo, se não receber significa que não está activo. Também se consegue este tipo de informação usando outras funcionalidades do protocolo ICMP, tais como, mensagens ICMP tipo 13 (*TIMESTAMP*) e ICMP tipo 17 (*ADDRESS MASK REQUEST*). Existem ainda *TCP sweeps* e *UDP sweeps* que funcionam de forma semelhante ao *ICMP Sweep* [17].

Depois de conhecer que sistemas estão activos, o próximo passo é determinar que serviços estão a correr ou activos no sistema alvo, efectuando ligações aos portos TCP e UDP do sistema. Temos então um *Port scanning*.

Diferentes processos podem ser usados para efectuar um *Port-Scan*, entre os quais:

- *TCP connect scan*: faz uso do mecanismo básico para o estabelecimento de ligações TCP; sendo enviado um pacote com a *flag* SYN activa para um porto. Na resposta, se for recebido um pacote com as *flags* SYN/ACK significa que o porto em questão está activo. Se as *flags* RST/ACK estiverem activas, então o porto não está activo e a ligação sofre um *reset*. No caso do porto estar activo é enviado um pacote com a *flag* ACK activa e a ligação é terminada após o processo completo de ligação ter sido feito.

- *TCP SYN Scan* (*half open scanning*): este tipo de *scan* difere do anterior pois não é efectuada uma ligação TCP total. Um pacote com a *flag* SYN é enviado para iniciar a ligação e espera resposta. Se um pacote com as *flags* SYN/ACK é recebido, temos indicação que o porto alvo está activo. Se as *flags* recebidas forem RST/ACK temos indicação que o porto não esta activo. No caso do porto estar activo a

ligação é imediatamente terminada enviando um pacote com a *flag* RST activa.

Outros *half open scanning* podem ser feitos usando outras combinações de *flags* [17].

2.3.1.3 – Compromisses

Este tipo de ataques usa vulnerabilidades conhecidas tais como *buffer overflows* e pontos fracos na segurança para forçar a entrada nos sistemas e assim obter acesso privilegiado aos *host*s. Dependendo da fonte (a partir do interior ou do exterior da rede), este tipo de ataques pode-se dividir nas duas seguintes categorias:

- **R2L** (*remote to local*) – aqui o atacante tem a capacidade de enviar pacotes para uma máquina na rede (sem que tenha uma conta no sistema), ganhando assim acesso (como *user* ou como *root*) a essa máquina. Na maior parte dos ataques R2L, o atacante invade a rede através da Internet. Exemplos típicos deste tipo de ataque incluem adivinhação de *passwords* e ganho de acesso através da exploração de vulnerabilidades no software.
- **U2R** (*User to root*) – neste caso um atacante que tem uma conta no sistema é capaz de fazer um uso indevido ou elevar os seus privilégios no sistema através da exploração de vulnerabilidades na rede, *bugs* no sistema operativo ou num programa instalado no sistema. Ao contrário dos ataques R2L, onde o atacante entra no sistema a partir do exterior, nos ataques U2R o atacante já se encontra no sistema e tipicamente torna-se *root* ou um utilizador com privilégios mais elevados. O ataque U2R mais comum é o *buffer overflow,* no qual o atacante explora erros de programação e tenta armazenar mais dados num *buffer* do que este é capaz de

suportar. Como os *buffers* têm capacidade limitada, os dados adicionais criados pelo atacante vão encher os *buffers*, corrompendo ou fazendo perder os dados válidos do sistema [7]. Por exemplo, uma vulnerabilidade descoberta no Microsoft Outlook e Outlook Express em Julho de 2000 permitia aos atacantes, simplesmente enviando uma mensagem de e-mail, provocar um *overflow* com dados supérfluos em determinadas áreas [18].

2.3.1.4 – Vírus, *Worms*, *Trojan horses*

Podemos definir vírus, *worms*, e *trojan horses* como programas que se replicam através da rede. Não existe um consenso quanto à definição precisa destes três termos, contudo o mais comum é que *trojan horses* são programas que efectuam algo de malicioso, como captura de *passwords*, quando executados por utilizadores insuspeitos; *worm* é algo que se replica; e vírus são *worms* que se replicam colando-se a outros programas [12]. De forma mais precisa:

- **Vírus**: código criado com o intuito expresso de se duplicar. Um vírus anexa-se a um programa anfitrião e, de seguida, tenta espalhar-se de computador em computador. Pode danificar equipamento, software, ou informações [19].

  Tal como a gravidade dos vírus humanos varia entre uma gripe de 24 horas e o efeito do Ébola, o impacto dos vírus informáticos varia entre um efeito ligeiramente inconveniente (como inofensivas mas irritantes mensagens no ecrã) e um elevado grau de destruição (como por exemplo apagar dados). Mas um verdadeiro vírus informático, regra geral, não se espalha sem interacção humana, tal como abrir um anexo enviado por e-mail, para que se possam replicar e propagar.

Os vírus podem ser classificados através das suas características em quatro áreas principais: ambiente, sistema operativo, algoritmo e nível de ataque [20]. Contudo, a classificação não é fácil, uma vez que muitos vírus possuem múltiplas características e podem inserir-se em várias das categorias citadas. Existem também outras classificações baseadas em como é o que os vírus infectam os sistemas.

Em 2000 havia mais de 50,000 vírus informáticos e a Symantec em Abril de 2008 apontou para um número superior a um milhão [21]. A mesma Symantec criou 2,895,802 novas assinaturas de códigos maliciosos em 2009, o que significa um incremento de 71% face a 2008 [22].

- ***Worms:*** Uma subclasse de vírus. Um *worm* normalmente espalha-se sem interacção por parte do utilizador e distribui cópias completas (possivelmente modificadas) de si próprio através das redes. Um *worm* pode consumir memória ou largura de banda, o que pode fazer com que um computador fique bloqueado [19].

  Tal como um vírus, replica-se mas de forma automática e agressiva através da rede, tirando partido das funções automáticas de envio e recepção de pacotes. Em primeiro lugar, toma controlo de funções do computador que permitem transportar ficheiros ou informações. O *worm*, após ter entrado no sistema, pode movimentar-se sozinho. Um dos grandes perigos dos *worms* é o facto de se poderem duplicar em grande volume. Por exemplo, um *worm* pode enviar cópias de si próprio para todas as pessoas que estejam no seu livro de endereços de correio electrónico, e os computadores dessas pessoas farão o mesmo, causando um efeito de avalanche o que pode resultar em congestionamentos nas redes das empresas e em toda a Internet. Quando são libertados novos *worms,* estes espalham-se bastante depressa. Entopem as redes e podem criar grandes períodos de

espera para abrir páginas na Internet. Como os *worms* não precisam de se propagar através de um programa ou ficheiro "hospedeiro", podem infiltrar-se no sistema do utilizador e permitir que outra pessoa possa assumir o controlo do seu computador à distância.

Worms podem ser divididos em varias categorias [7]:

i. *Worms* tradicionais;
ii. *Worms* de e-mail;
iii. *Worms* de partilha de ficheiros Windows;
iv. *Worms* híbridos.

Alguns *worms* têm sido usados para lançar ataques DoS, como por exemplo os *worms erkms* e *li0n* que foram usados para provocar um DDoS através de vulnerabilidades do BIND [7]. Outro exemplo é o *worm* SQL Slammer que foi lançado na Internet no dia 25 de Janeiro de 2003; a partir do momento do seu lançamento este *worm* alcançou toda a rede mundial em aproximadamente dez minutos, comprometendo muitos sistemas e incapacitando cinco dos treze principais servidores DNS de então [23, 24].

- ***Trojan Horses***: "Cavalo de Tróia" – Programa de computador que aparenta ser útil, mas na verdade causa danos [19].

  O cavalo de Tróia da mitologia aparentava ser uma prenda, mas na realidade continha no seu interior soldados gregos que se apoderaram da cidade de Tróia. Do mesmo modo, os *trojan horses*, modernos cavalos de Tróia, são programas de computador que aparentam ser software útil mas na realidade comprometem a segurança do utilizador e causam muitos danos. A sua propagação e feita quando os utilizadores, inadvertidamente, executam um programa que pensam ser legítimo ou de fonte segura.

A distinção entre vírus, *worms* e *trojan horses* não é fácil. Tomemos como exemplo o *'Love Bug'* [25], que em 2000 infectou milhões de computadores no mundo inteiro. Trata-se de forma simultânea de um vírus, *worm* e *trojan*. *Trojan horse* pois pretendia ser uma carta de amor mas era de facto um programa malicioso. Vírus porque infectava outros ficheiros no disco tornando-os em novos *trojan horses*. Finalmente *worm* pois propagava-se de forma automática e autónoma através da Internet.

2.3.2 – Número de ligações envolvidas no ataque

Os ataques podem ser classificados de acordo com o número de ligações envolvidas no ataque:

- **Múltiplas ligações**: são efectuadas múltiplas ligações para realizar o ataque (figura 7). Exemplos típicos desta variante são os ataques DoS, *probing* e *worms.*

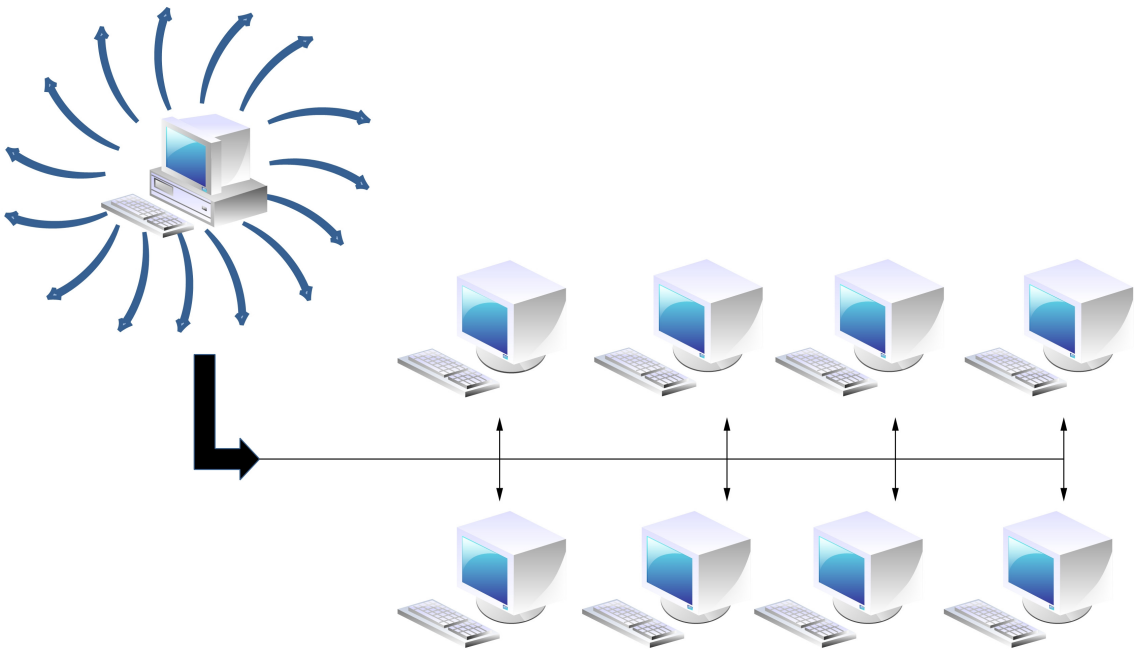

Figura 7 – Exemplo de um ataque de múltiplas conexões

- **Muito poucas ligações ou ligação única:** ao contrário do caso anterior, apenas uma ligação ou um número reduzido de ligações são efectuadas para levar a cabo o ataque. É exemplo deste tipo os *buffer overflows*.

2.3.3 – Fonte do ataque

Neste ponto é avaliada a origem do ataque que pode ser lançado a partir de uma única localização (fonte de ataque única), sendo este o caso mais comum e típico de ataques do tipo *scanning*, ou a partir de várias localizações (ataque distribuído ou coordenado), sendo esta topologia usada em ataques do tipo DDoS ou outros ataques organizados.

No caso dos ataques distribuídos é comum que o alvo destino não seja também único mas sim múltiplos alvos em diferentes destinos.

2.3.4 – Ambiente

Os ataques podem ser categorizados tendo em conta o ambiente onde ocorrem:

- **Intrusões no *host***: intrusão que ocorre numa máquina específica, que pode ou não estar ligada à rede. A detecção destas intrusões é normalmente feita investigando as informações do sistema, tais como *logs* do sistema. A identificação do atacante está normalmente associada à utilização de um *username* e é portanto relativamente fácil de descobrir.

- **Intrusões na rede**: intrusões que ocorrem através da rede, normalmente feitas a partir de fora da rede. A sua detecção é feita através da análise dos dados de tráfego da rede. Esta análise permite a detecção do ataque mas não é precisa quanto à identificação do atacante.

- **Intrusões num ambiente P2P (*pear to pear*):** são intrusões que ocorrem num sistema onde os computadores actuam como *peers* na Internet. Ao contrário da arquitectura cliente/servidor, em ambientes P2P os computadores têm capacidades e responsabilidades equivalentes e não têm endereços IP fixos. Estão normalmente localizados nas margens da Internet e desligados dos sistemas DNS [26].

  Embora aplicações de partilha de ficheiros P2P possam aumentar a produtividade das redes empresariais, também podem apresentar vulnerabilidades, uma vez que permitem aos utilizadores efectuar downloads de códigos executáveis que podem introduzir aplicações de *backdoor* indetectáveis nas máquinas dos utilizadores e assim comprometer a segurança de toda a rede.
- **Intrusões em redes *wireless*:** tratam-se de intrusões que ocorrem entre computadores ligados através de redes *wireless*. A detecção de ataques em redes *wireless* é baseada na análise de informações acerca das ligações nas redes *wireless,* normalmente recolhidas nos pontos de acesso. As ameaças às redes *wireless* podem ser categorizadas em:

    - *Eavesdropping* (escutas): quando o atacante apenas está à escuta de dados.
    - Intrusões**:** quando o atacante tenta aceder ou modificar dados.
    - *Communication hijacking* (sequestro de comunicações)*:* quando é criado um ponto de acesso

falso ou desonesto que atrai nós móveis a si para a eles se ligar e recolher dados confidenciais.

- *Denial of Service (Jamming)***:** quando um atacante perturba o canal de comunicação com variações na frequência ou obstáculos físicos e assim desactiva todas as comunicações no canal.

2.3.5 – Nível de automatização

Dependente do nível de automatização do ataque, temos diferentes categorias:

- **Ataques automáticos:** usam ferramentas automatizadas que são capazes de sondar e fazer *scan* a uma grande parte da Internet num curto período de tempo. Usando estas ferramentas, facilmente disponíveis, mesmo atacantes inexperientes podem desencadear ataques altamente sofisticados. Este tipo de ataque é provavelmente o método mais comum de atacar sistemas informáticos nos dias de hoje.

- **Semi-automáticos:** utilizam scripts automatizados para fazer a verificação e comprometer máquinas na rede e efectuar a instalação de código de ataque. De seguida, especificam o tipo de ataque e o endereço da(s) vítimas(s).

- **Ataques manuais:** envolvem um *scan* manual das máquinas e geralmente requerem grande conhecimento e trabalho. Não é um tipo de ataque frequente, mas é normalmente mais perigoso e difícil de detectar do que os restantes dois. Grupos organizados de atacantes usam, geralmente, este tipo de ataque para atacar sistemas de importância crítica.

2.4 – IDS (*Intrusion Detection Systems*)

IDS são softwares que têm como função detectar, identificar e responder a actividades anormais e não autorizadas num sistema. O objectivo de um IDS é estabelecer um mecanismo para detecção de violações de segurança. As violações podem ser iniciadas por pessoas de fora da rede, que tentam entrar no sistema, ou por pessoas dentro da rede, que tentam fazer um uso ilegítimo de seus privilégios. Os IDS recolhem informações de uma variedade de sistemas e fontes na rede para então analisar as informações e procurar sinais de intrusão.

Em 1987, Dorothy Denning [27] desenvolveu o primeiro modelo genérico de detecção de intrusão de que se tem conhecimento, independente de qualquer sistema em particular, vulnerabilidade de sistema ou tipo de intrusão. Este modelo utilizava dados estatísticos, gerados a partir de registos de auditoria, e baseava-se na hipótese de que violações de segurança podiam ser detectadas através da monitorização desses registos e na identificação de padrões anormais de uso do sistema.

Desde o aparecimento deste modelo, muitos IDS foram propostos tanto no âmbito académico e de pesquisa como no âmbito comercial. Apesar de muito diferentes entre si nas técnicas usadas para recolha e análise de dados, a grande maioria sistemas de detecção de intrusão obedecem a uma arquitectura (figura 8), composta pelos seguintes componentes:

- **Recolha de Dados *(Data gathering, sensors)*:** responsável pela recolha de dados do sistema monitorizado;
- **Detector:** efectua o processamento dos dados recolhidos com o objectivo de detectar intrusões;

- **Base de dados *(Knowledge base)*:** contém informações recolhidas pelos sensores (*data gathering*), mas pré-processadas (por exemplo bases de ataques e as suas assinaturas). Estas informações são normalmente fornecidas por especialistas em redes e segurança de redes;
- **Configuração *(Configuration)*:** dispositivo que fornece informações sobre o estado actual do sistema de detecção de intrusão;
- **Componente de Resposta (*Response component*):** componente que inicia acções de resposta quando é detectada uma intrusão. As respostas podem ser automatizadas (activo) ou envolveram acção humana (inactivo).

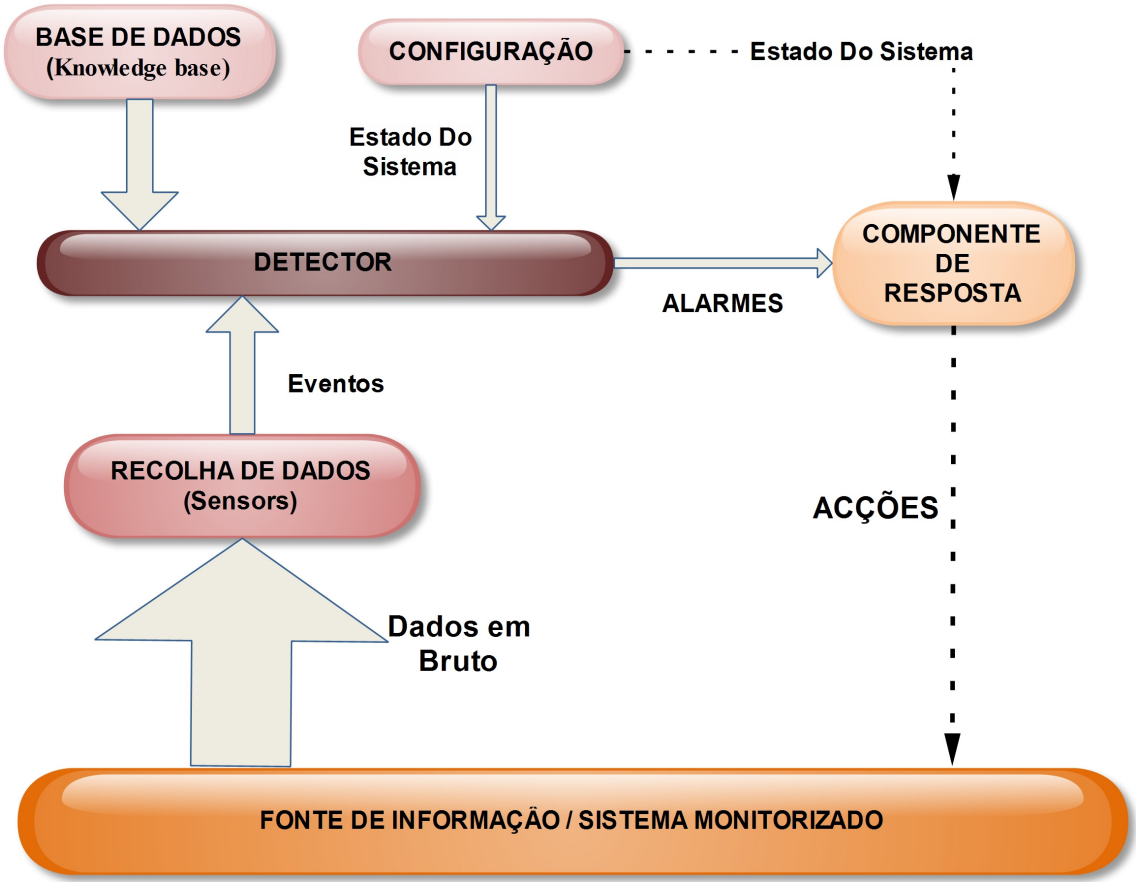

Figura 8 – Arquitectura básica de um IDS

2.4.1 - Características de um IDS

Um dos problemas enfrentados pelos sistemas de detecção de intrusão, e que prejudicam também o seu desempenho, prende-se com a real natureza dos eventos detectados, do ponto de vista da segurança, e da classificação realizada pelo detector. Assim, encontra-se na figura 9 o resumo dos quatro casos possíveis. *True negative* (verdadeiros negativos) e *true positive* (verdadeiros positivos) correspondem a um correcto funcionamento do detector, ou seja, eventos que são inofensivos e classificados como normais e ataques que são classificados como tal. Os *false positive* (falsos positivos) e *false negative* (falsos negativos) são os eventos que prejudicam a detecção quando a hipótese suspeita não é confirmada. Os falsos positivos também conhecidos como falsos alarmes são de facto o factor limitador dos IDSs.

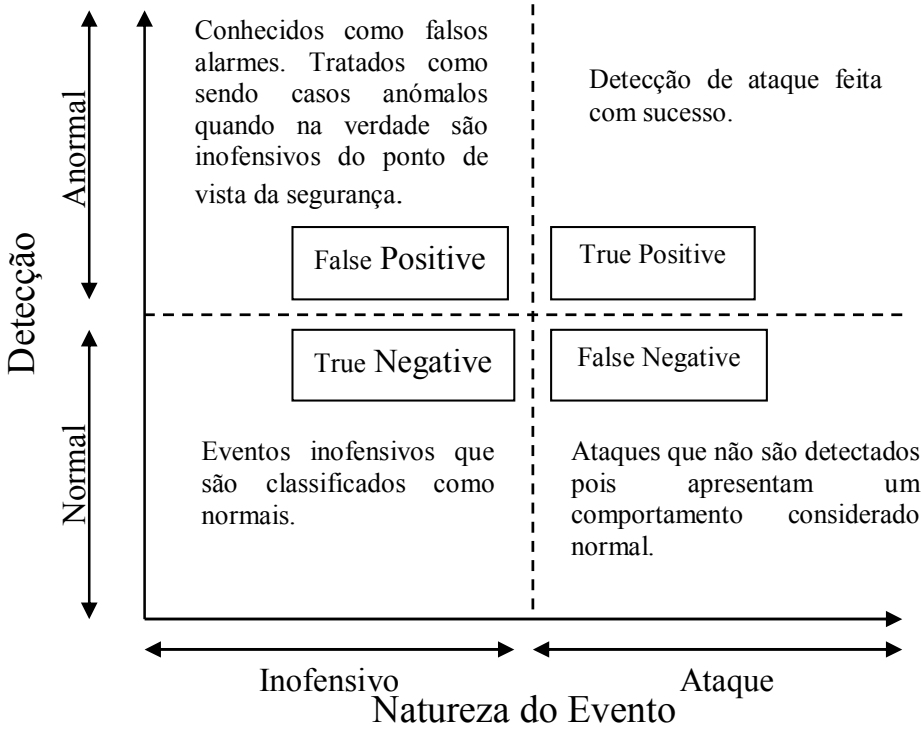

Figura 9 – Possíveis casos de detecção

(adaptado de *Anomaly Detection Methods*, [28])

2.4.2 – Eficiência de um IDS

Um sistema de detecção de intrusão deve reunir uma série de características que permitem avaliar a sua eficiência [29]:

- Precisão: imprecisão ocorre quando um IDS assinala uma acção legítima como anómala ou intrusiva;
- Desempenho: o desempenho de um IDS é a taxa de processamento dos eventos. Se esta taxa for baixa a detecção em tempo real torna-se impossível e a detecção sem ser em tempo real torna-se lenta, podendo ser já tarde demais quando um evento é assinalado como malicioso;
- Plenitude (*completeness*): falta de plenitude ocorre quando um IDS falha na detecção de um ataque. Esta característica é bastante mais difícil de avaliar que as restantes, pois é impossível ter um conhecimento global acerca dos ataques ou abusos de privilégios.
- Tolerância de falhas: um IDS deve ele próprio ser resistente a intrusões, particularmente do tipo DoS e deve de ser desenhado tendo este objectivo em conta. Esta característica adquire particular importância pois a grande maioria dos sistemas de detecção de intrusão correm sobre sistemas operativos ou hardware de uso comercial, que se sabe serem vulneráveis a ataques.
- Oportunidade/tempo de resposta (*Timeliness*): um IDS tem de realizar e comunicar a sua análise o mais rápido possível para que seja possível uma reacção rápida antes que demasiados estragos sejam feitos, e também para prevenir o atacante de eliminar o seu rasto aos olhos do gestor de rede ou mesmo do IDS. Esta característica toma maior importância que o desempenho porque abrange não só a

velocidade de processamento intrínseca mas também o tempo necessário para propagar a informação e reagir a ela.

2.5 – Taxonomia de um IDS

Existe uma série de conceitos e características usada para a classificação de sistemas de detecção de intrusões, tendo sido propostas ao longo dos anos várias classificações diferentes. Contudo, como no caso dos ataques, não existe uma taxonomia universalmente aceite.

Será apresentada uma taxonomia baseada na síntese de outras taxonomias propostas e baseada na recolha feita por A. Lazarevic, V. Kumar e J. Srivastava [7].

São adoptados cinco critérios, tal como apresentado na figura 10. Cabe ressaltar que em geral os IDSs usam exclusivamente uma técnica de detecção e uma origem de dados. No entanto, existem pesquisas que mostram as vantagens de abordagens híbridas, como no caso de IDSs baseados em anomalias com geração automática de assinaturas ou assinaturas pré-programadas e IDSs com múltiplas origens de dados [30-32].

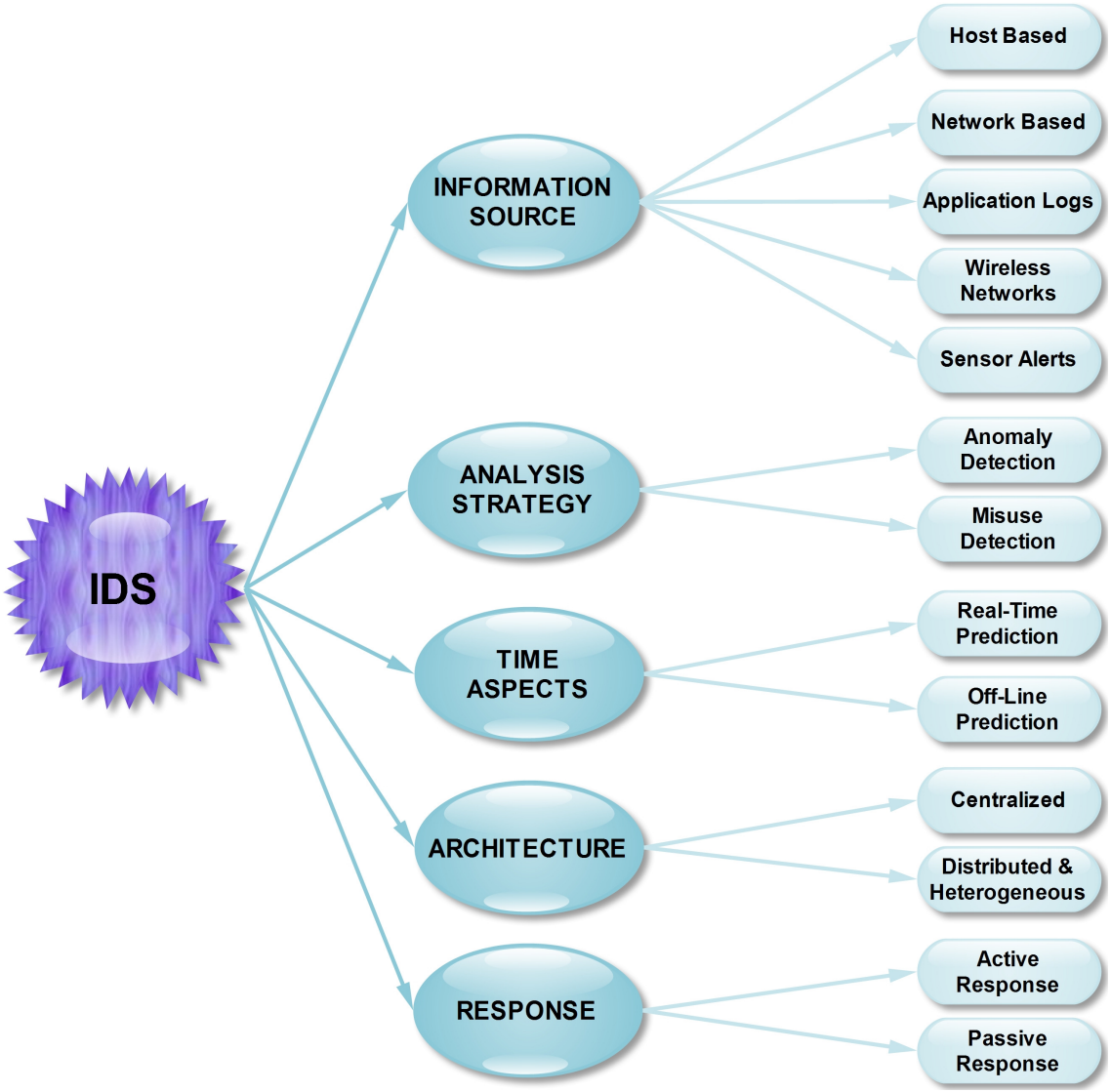

Figura 10 – Taxonomia de um IDS

(Adaptado de *Managing cyber threats: issues, approaches, and challenges* [7]

2.5.1 – *Information Source*

Trata-se da categorização de um IDS tendo em conta a fonte dos dados que são analisados. É feita uma subdivisão desta categoria em cinco pontos, sendo as fontes de informação mais comuns e mais vezes referenciadas a *Host based*, *Network based* e *application logs*. Estes três tipos serão brevemente explicados de seguida. Para um

conhecimento mais profundo destes três tipos e dos dois restantes aconselha-se a leitura da seguinte bibliografia: [3, 7, 29, 33]

2.5.1.1 – *Host-based* IDS (HIDS)

Sistemas de detecção de intrusão baseados no *host* foram os primeiros a ser explorados na área de detecção de intrusões. O seu principal objectivo é monitorizar toda a actividade existente num determinado *host*. O funcionamento destes sistemas baseia-se na recolha e análise de pacotes originados num *host* que tem um determinado serviço a correr. Depois de recolhidos, esses dados podem ser analisados localmente ou até enviados para uma máquina remota responsável pelo exame.

Sistemas HIDS utilizam os registos de auditoria como principal fonte de informação para realizar a detecção[7, 29, 33].

Aqui as informações relevantes são tempos de uso do CPU, número de acessos ao sistema via terminal remoto, tentativas de login, utilização de recursos do sistema, etc.

As principais vantagens dessa abordagem são [3]:

a) Independência de rede: independentemente da forma de comunicação utilizada entre os *hosts,* as tarefas de um IDS baseado no *host* não são directamente afectadas;

b) Detecção de ataques internos: é mais fácil para um IDS baseado no *host* detectar actividades não autorizadas que representem abusos de privilégio por parte de utilizadores ou programas;

c) Reacção: embora não sendo uma actividade de responsabilidade directa do IDS, pode-se, com maior eficiência e facilidade, avaliar danos e recuperar erros usando uma ferramenta baseada no *host*.

Como desvantagens realçam-se os seguintes pontos [3]:

a) Dificuldade de instalação: cada *host* monitorizado deve conter pelo menos um elemento do IDS baseado em *host* instalado localmente, dificultando a instalação;
b) Dificuldade de manutenção: pelo mesmo motivo apresentado acima, a tarefa de manutenção destas ferramentas é dificultada;
c) Ataques ao próprio IDS: como os elementos da detecção devem estar localmente instalados, um atacante que conseguir invadir tal *host* pode desabilitar ou destruir a ferramenta instalada;
d) Dificuldade de tratar ataques de rede: alguns ataques são especialmente direccionados à infra-estrutura de rede, sendo dificilmente tratados por HIDS;
e) Desempenho: ferramentas desse tipo são extremamente intrusivas, ou seja, interferem directamente no funcionamento e desempenho do sistema monitorizado;
f) Dependência de plataforma: um IDS com estas características é altamente dependente da plataforma de monitorização, devendo sofrer muitas modificações para se adaptar a outros ambientes.

2.5.1.2 – *Network-based* IDS (NIDS)

Com o rápido crescimento e popularidade da Internet o número de ataques dirigidos à própria rede aumentou consideravelmente. Alguns destes ataques tem a particularidade de não serem detectados pela análise única dos *host*s. Por estas razões, desenvolveram-se ferramentas capazes de analisar o tráfego de rede com vista À detecção de ataques. A maioria dos sistemas IDS existentes no mercado é do tipo NIDS [7, 29].

IDSs baseados em Rede monitorizam toda a informação que passa através da rede. É feita uma análise de cada um dos pacotes que são capturados para averiguar se estão dentro dos padrões que foram pré-determinados ou não, indicando assim que se trata de um tráfego normal ou de uma tentativa de ataque.

Um NIDS precisa de trabalhar em modo promíscuo para fazer a análise de pacotes não destinados a ele. Tendo todas as informações que transitam pela rede é possível ao sistema, por exemplo, detectar *port scans*, monitorizar ligações ou datagramas maliciosos.

As principais fontes de informação de um NIDS são as informações obtidas através de variáveis do protocolo SNMP e através de informações recolhidas dos pacotes de rede, como sejam no caso dos pacotes TCP/IP, o cabeçalho e conteúdo dos pacotes, entre outras.

As principais vantagens desse tipo de IDS são [3]:

a) Detecção de ataques externos: é mais fácil, para um IDS baseado em rede, detectar actividades não autorizadas desencadeadas por utilizadores ou programas externos;
b) Facilidade de instalação: desde que correctamente distribuídos, poucos detectores baseados em rede podem monitorizar todas as actividades de uma rede de grandes dimensões;
c) Facilidade de uso: pelo mesmo motivo acima, é mais fácil manter actualizado um IDS baseado em rede;
d) Desempenho: a instalação de IDSs de rede representa um impacto muito pequeno na rede existente. Compostos, na sua maioria, por dispositivos passivos, esse tipo de ferramenta praticamente não interfere no funcionamento normal do sistema;
e) Independência de plataforma: como os alvos são os dados recolhidos directamente na rede, a sua utilização é praticamente independente das plataformas monitorizadas.

Entre as suas principais desvantagens, estão [3]:

a) Tratamento de redes de alta velocidade: os NIDS apresentam muitas dificuldades no tratamento de grandes quantidades de dados. Com a popularização de redes cada vez mais rápidas, isso tende a tornar-se um problema cada vez mais sério;
b) Dependência de rede: alterações na infra-estrutura de rede possuem reflexos significativos. Com a utilização de elementos de rede como *switches*, por exemplo, a tarefa de capturar pacotes fica prejudicada;
c) Dificuldade de reacção: a reacção a ataques em curso é, muitas vezes, impossível.

Devido aos regulamentos de privacidade que são cada vez mais comuns, a monitorização de redes tem de ser cuidadosamente pensada de forma a estar de acordo com as exigências legais e locais para tal actividade [33].

2.5.1.3 – *Application logs*

Application logs pode ser considerado como uma subdivisão do *host based* IDS. Monitorizam apenas aplicações específicas. Os *application based* IDS têm acesso a tipos de informação que não são obtidos quer por NIDS quer por HIDS. A fonte de informação mais comum usada por este tipo de IDS são os *log files* das aplicações [3, 7].

2.5.2 – Estratégia de análise

Existem duas aproximações possíveis para a análise de eventos destinada a detectar ataques: detecção de intrusão por abuso

(*misuse detection*) e detecção de intrusão por anomalia (*anomaly detection*), sendo que existem vários IDS que integram estas duas abordagens com o objectivo de beneficiar das vantagens de ambas.

2.5.2.1 – Detecção de intrusão por abuso (*misuse detection*)

A detecção de intrusão por abuso é baseada na ideia que os ataques podem ser representados na forma de padrões ou assinaturas, de tal forma que até variações de um mesmo ataque podem ser detectadas, e exige um elevado grau de conhecimento dos ataques e vulnerabilidades dos sistemas. Trata-se da abordagem mais comum nos IDS comerciais e apesar de ser bastante eficaz na detecção de ataques já conhecidos, que deixam traços característicos, é bastante ineficaz na detecção de novos ataques. A figura 11 mostra a representação básica de um sistema de detecção de intrusão por abuso.

Kumar [7] faz a divisão de detecção de intrusão por abuso em 4 categorias:

- ***Signature based tecnhiques*** (técnicas baseadas em assinatura): funcionam procurando numa base de dados, por uma assinatura (sequências de acções que seriam consideradas indícios de uma intrusão), característica de cada ataque. A base de dados tem de ser actualizada para cada novo tipo de ataque que se descubra. Como exemplos de *signature based* IDS conhecidos temos o SNORT [34] e o Haystack [35, 36].

- ***Rule-based techniques*** (sistemas especialistas, sistemas baseados em regras): o conhecimento sobre os ataques é codificado na forma de regras de condições do tipo *if-then*. Um dos grandes problemas desta abordagem é a imposição de uma sequência determinada por regras, nem sempre verificada em dados reais, ou seja, alguns eventos podem ocorrer numa ordem pré-estabelecida, passando assim despercebidos pela avaliação feita.
- ***State-transition analysis*** (análise por transição de estados): os ataques são representados como mudanças de estado no sistema monitorizado. Sucessivos estados são ligados entre si por arcos que representam os eventos/condições necessários à mudança de estado. O tempo dispendido na análise de cada transição é a principal desvantagem destes sistemas.
- ***Data mining based tecnhiques***: cada instância num conjunto de dados é rotulada como sendo *normal* ou intrusiva e um algoritmo inteligente é treinado a partir dos dados rotulados.

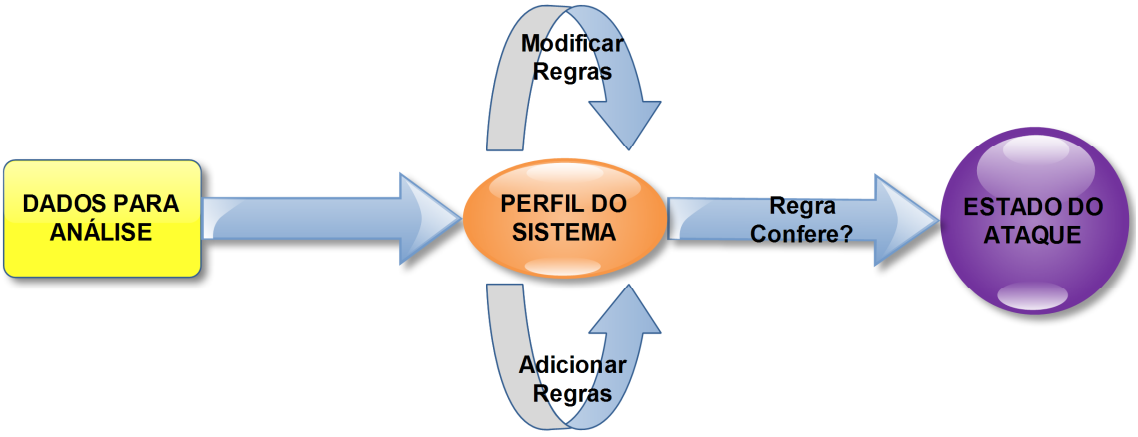

Figura 11 – Esquema básico de um IDS por Abuso

(Adaptado de *An introduction to Intrusion Detection* [37])

2.5.2.2 – Detecção de intrusão por anomalia (*anomaly detection*)

A detecção de intrusão por anomalia assenta na ideia de que todo o evento intrusivo é necessariamente anómalo. Um intruso, ao invadir um sistema, não conhece o padrão de uso dos recursos, gerando assim, um comportamento anómalo. Por comparação constante do perfil do estado actual do sistema com o perfil do estado, criado inicialmente, do sistema em estado normal, antes de alguma intrusão, é possível procurar indícios de mudanças de comportamento, podendo estas mudanças ser sinónimas de ataques. Na mudança destes perfis do sistema reside um dos grandes problemas deste método de detecção, pois gera um elevado número de falsos positivos. A criação e manutenção dos perfis actualizados e sob vigilância constante têm um custo elevado, sendo esta outra das desvantagens deste método [38].

O crescente número de ataques, a sua severidade e complexidade aumentou consideravelmente o interesse em algoritmos de detecção de intrusão por anomalia, devido à sua capacidade de reconhecer novas ameaças [7].

A figura 12 mostra a representação básica de um sistema de detecção de intrusão por anomalia.

Kumar [7] faz a divisão de detecção de intrusão por anomalia em 5 categorias:

- ***Statical methods*** (métodos estatísticos): o comportamento do utilizador ou do sistema é monitorizado, medindo várias variáveis ao longo do tempo. Os períodos de amostragem podem variar de alguns minutos até meses. Desvios destes valores quando comparados com situações normais indiciam

uma situação anómala. Alguns IDS conhecidos usam esta abordagem: IDES [39], NIDES [40], EMERALD [31].

- ***Rule based methods*** (métodos baseados em regras): sistemas *Rule Based* usados na detecção de intrusões caracterizam o comportamento normal dos utilizadores, redes e/ou sistemas através de um conjunto de regras.

- ***Distance based methods***: a maioria das abordagens estatísticas tem limitações na detecção de *outliers*[1] em espaços de maior dimensão. Os métodos baseados na distância tentam ultrapassar esta limitação fazendo a detecção de *outliers* com base no cálculo das distâncias entre os pontos.

- ***Profiling methods***: neste caso perfis de comportamento considerados normais são construídos para diferentes tipos de tráfego, utilizadores, programas, etc., e desvios destes perfis são considerados como tentativas de intrusão.

- ***Model based methods***: vários investigadores usam diferentes tipos de modelos para caracterizar o *normal* comportamento *normal* dos sistemas monitorizados. No método baseado em modelos, anomalias são detectadas como sendo desvios do modelo que representa o comportamento *normal*.

[1] Outliers – Observações que apresentam um grande afastamento das restantes ou são inconstantes com elas. Estas observações também são designadas por observações anómalas, contaminantes ou aberrantes.

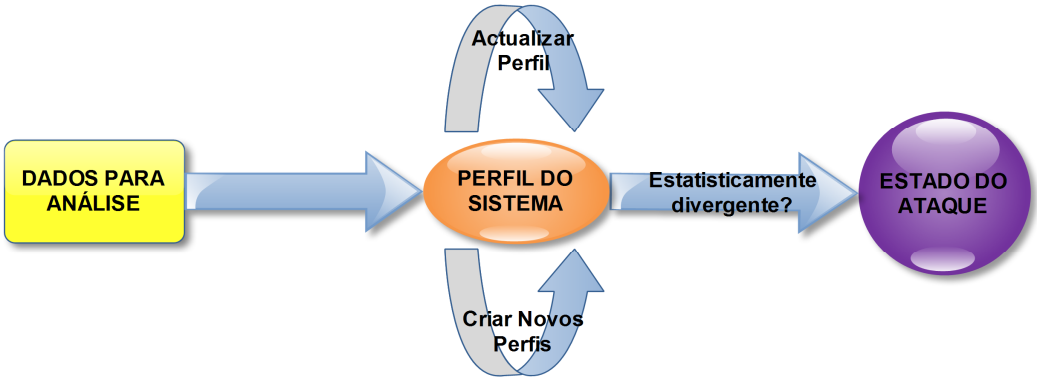

Figura 12 – Esquema básico de um IDS por Anomalia

(Adaptado de *An introduction to Instrusion Detection* [37])

2.5.3 – Aspectos temporais

Considerando os aspectos temporais, os IDSs podem ser divididos em dois grandes grupos: IDS em tempo real e IDS *off-line*, por vezes também referidos como monitorização contínua e monitorização periódica, respectivamente.

No primeiro grupo, os IDS tentam fazer uma detecção em tempo real ou próxima do tempo real, funcionando com fontes de dados contínuas, recolhendo os dados logo após a sua geração, analisando-os enquanto recebe e processa outros dados, tudo isto sem interromper o *normal* funcionamento do sistema monitorizado.

Já os IDS *off-line* efectuam uma análise dos dados à posteriori. Estas análises off-line são muitas vezes efectuadas através de ferramentas de análise estática, que periodicamente efectuam recolha de dados/*snapshots* no ambiente monitorizado, fazendo depois uma análise para procurar ataques. Embora estas ferramentas sejam muito populares e amplamente usadas por gestores de rede, não são tipicamente suficientes para garantir altos níveis de segurança [7, 29].

Como exemplos deste tipo de configuração podemos citar o COPS [41] e o Tiger [42].

2.5.4 – Arquitectura

A recolha de dados e a respectiva análise podem ser feitas de duas formas distintas: de forma centralizada e de forma distribuída.

Na forma centralizada a análise dos dados é realizada num número fixo de localizações, independentemente do número de *host* que são monitorizados. Na forma distribuída a análise dos dados é realizada num número de localizações que é proporcional ao número de *hosts* que estão a ser monitorizados [43].

A maioria dos IDS emprega estratégias de forma centralizada, detectando intrusões que ocorrem num único sistema monitorizado. Contudo, o aumento da tendência para ataques coordenados onde várias máquinas estão envolvidas, como atacantes ou como vítimas, tem aumentado o interesse na forma distribuída.

2.5.5 – Resposta

A partir do momento em que uma tentativa de intrusão é detectada, o IDS pode agir de forma passiva, enviando um alerta, ou de forma activa, despoletando alguma acção para bloquear a intrusão ou minimizar as suas consequências. Nos cenários mais comuns, o IDS tem respostas passivas e simplesmente notifica a ocorrência de um ataque, quer através de janelas de aviso quer através de registo dos ataques num ficheiro.

Um IDS ideal deveria automaticamente responder a um ataque sem necessidade de intervenção humana. Contudo tal trata-se de um cenário irrealista, principalmente devido à dificuldade de eliminar falsos alarmes [7].

3 – Metodologia

Em 2009, no âmbito de uma dissertação de Mestrado, Carlos Miranda [44] disponibilizou um conjunto abrangente de dados/capturas, correspondentes a diversos cenários de rede realistas, obtidos através da emulação em ambiente controlado de diferentes topologias, diferentes serviços e padrões de tráfego.

O núcleo da de emulação é composto pelo emulador GNS3 [45], que permite emular várias instâncias de IOS (Internetwork Operating System) de routers da CISCO e computadores reais que através de interfaces de rede virtuais (interfaces TAP) emulam a interacção de uma vasta gama de clientes com acesso aos diferentes serviços da rede, criando assim cenários reais de rede fechados, sem interferência do exterior.

O tráfego é criado através de scripts de geração de tráfego, desenvolvidos em linguagem *bash,* que permitem chamar os programas responsáveis pelo *download* e *upload* de tráfego. São utilizadas funções de probabilidade (como a distribuição exponencial) na geração de tráfego, contribuindo assim para o realismo dos cenários. Dos vários cenários emulados, usou-se neste trabalho o cenário B, descrito na figura 13

O tráfego é gerado num ambiente c*lean*, sem qualquer tipo de anomalia, e num ambiente de *port scan* ou seja, uma mistura de tráfego normal com ataques do tipo *port scan*. O cenário de *port scan* é dividido em três tipos, de acordo com o tempo de duração do ataque:

- *Sneaky*: usado para evasão aos IDS. *Port scan* muito lento, os *scans* não são feitos em modo paralelo e com um tempo de espera entre *probes* de 15 segundos;

- *Normal*: permite processos paralelos de *scan* sendo mais rápido que o *sneaky*;
- *Aggressive*: forma mais agressiva de *scan*. O intervalo entre *probes* não excede 10 milissegundos (para portos TCP).

Para cada tipo de ataque *port scan* usado é ainda feita uma divisão com base no número de máquinas envolvidas na fonte e na origem do ataque, como apresentado na tabela 1.

Tipo de ataque	Descrição
1 para 1	Um único atacante efectua um *port scan* a um único alvo.
1 para N	Um único atacante efectua um *port scan* a vários alvos. Normalmente isto é feito contra alguém com mais que um IP associado.
N para 1	Múltiplos atacantes contra um único alvo
N para N	Múltiplos atacantes contra vários alvos.

Tabela 1 – Ataques feitos para cada tipo de divisão temporal.

Para os ambientes de *port scan* 1 para N e *port scan* N para N, ambos no modo *aggressive*, não foi gerado qualquer tipo de tráfego, não sendo portanto possível a sua análise.

Os diversos tipos de *port scan* são do tipo TCP SYN *Scan* (*half open scanning*) [17] e foram feitos usando a ferramenta Nmap [46].

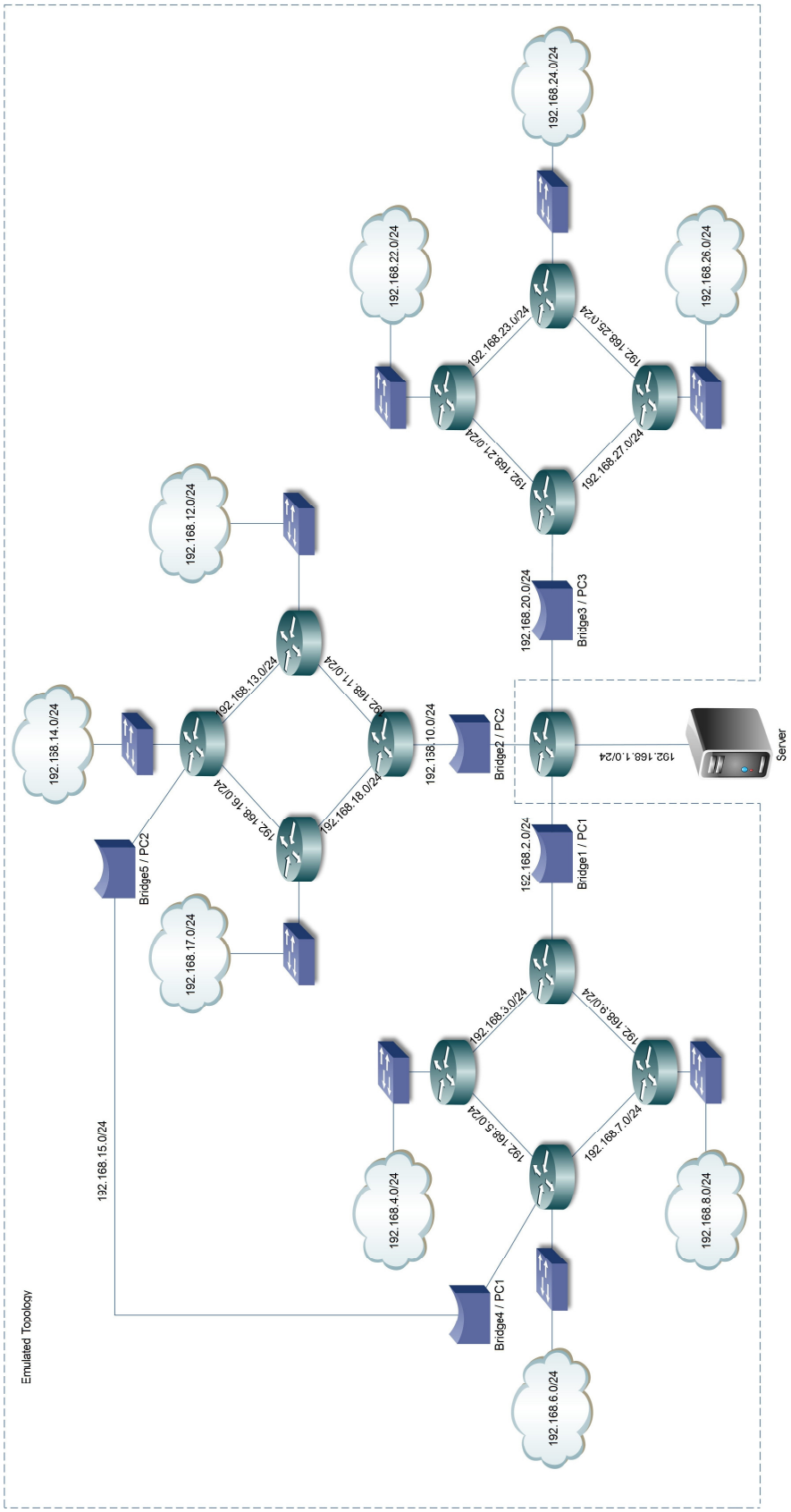

Figura 13 – Cenário de emulação usado

O tráfego gerado é guardado, em intervalos de 30 minutos, em ficheiros do tipo ".pcap" e neste trabalho são usadas apenas as capturas feitas no servidor, deixando de fora as capturas efectuadas nos clientes. Tal prende-se com o facto de que o tratamento das capturas nos clientes exige uma elevada capacidade de armazenamento de dados e os ataques gerados serem direccionados ao servidor.

Na figura 14 pode ver-se um esquema simplificado do processo de detecção.

A obtenção de dados é feita através do uso do Tshark a correr em Ubuntu versão 10.10 e a análise dos dados para obtenção de resultados é feita com recurso ao EasyFit e MatLab ambos a correr em Windows XP Professional Versão 2002 Service Pack 3.

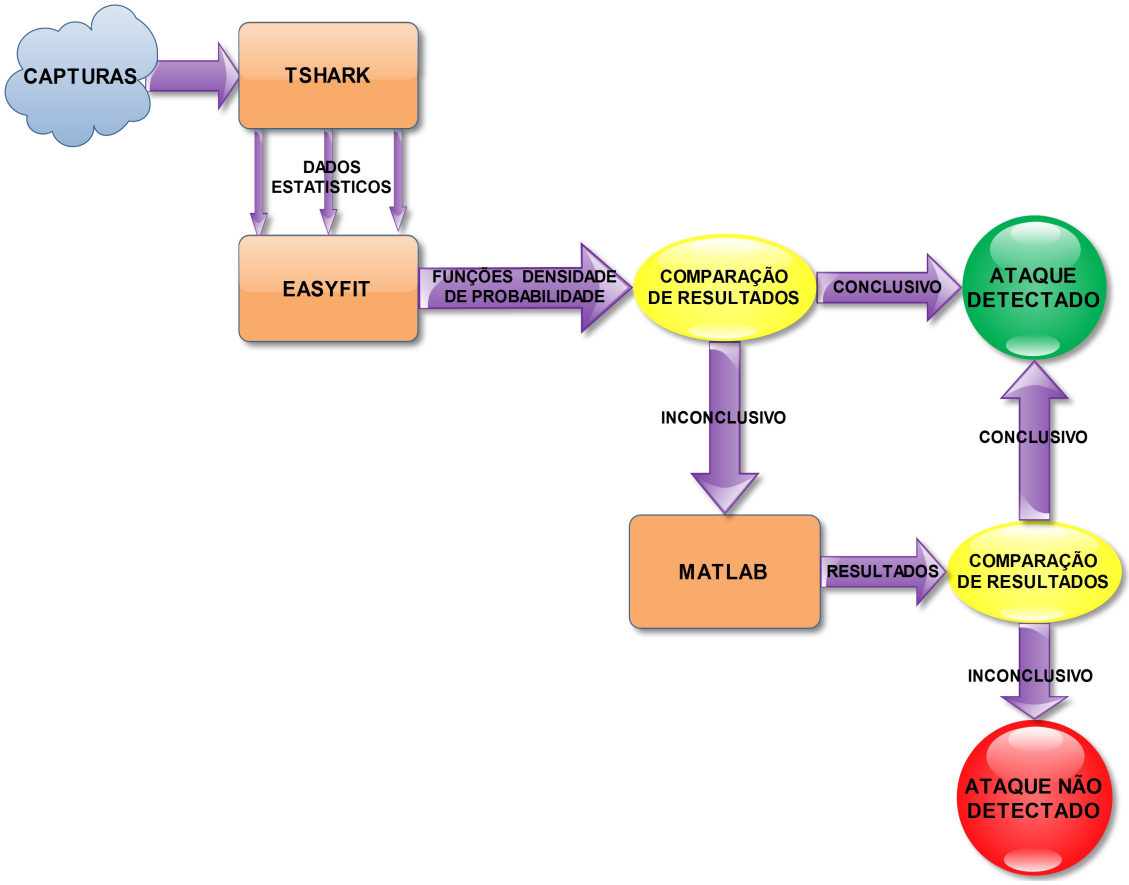

Figura 14 – Esquema simplificado de processo de detecção

3.1 – Tshark

Tshark [47] é uma ferramenta baseada em linha de comandos que faz parte do conhecido analisador de protocolos Wireshark [48], criado em 1998.

Tshark/Wireshark, anteriormente conhecido como ethereal é, à semelhança do tcpdump, baseado no libcap. No entanto, entre o Tshark e o tcpdump existe uma grande diferença a nível de desempenho. O Tshark usa um modelo multi-segmentos, com dois segmentos diferentes, um para captura de dados da rede e outro para processamento. Assim, consegue um melhor desempenho quando comparado com o tcpdump.

O uso da capacidade de descodificação ao nível do protocolo/aplicação do Tshark, partindo do conjunto de dados/capturas já referido, permite obter filtragens e dados estatísticos e exportá-los para formatos ".txt" com os quais é possível trabalhar de forma a obter características e padrões estatísticos das diferentes situações de rede.

Sendo o objectivo primário, tal como já foi referido, a detecção *de port scans*, usou-se o Tshark para obter os seguintes dados:

- ***Flags* SYN isoladas do protocolo TCP** (contagem do número de *flags* SYN isoladas por segundo, ou seja quando no octeto treze do protocolo TCP apenas a *flag* SYN se encontra activa). Estando esta *flag* envolvida no início das sessões TCP e sendo o *port scan* efectuado do tipo TCP SYN scan, a sua contagem adquire especial interesse. As *flags* SYN isoladas são sinónimo de início de sessão TCP, portanto o seu número por segundo indica o número de pedidos de início de sessões TCP por segundo.

- ***Flags* SYN** (contagem do número de *flags* SYN por segundo, ou seja, contagem de número de pacotes por segundo que contêm a *flag* SYN independentemente de esta ser a única *flag* activa ou não. A contagem de *flags* SYN por segundo quando estas não são as únicas activas no campo *code bits*, no octeto 13 do protocolo TCP indicam o número de respostas a pedidos de início de sessão.

- **Intervalo de tempo entre pacotes** (*frame time delta*). Dá a indicação do tempo que decorre entre a chegada de cada pacote.

- **Porto de origem quando só a *flag* SYN se encontra activa**. Os portos de origem quando só a *flag* SYN está activa dão indicação do porto a partir do qual é pedido um início de sessão.

- **Porto de destino quando só a *flag* SYN se encontra activa**. Os portos de destino quando só a *flag* SYN está activa dão indicação do porto a que se está a tentar aceder aquando do pedido de início de sessão TCP.

3.2 – EasyFit

EasyFit [49] na sua versão 5.4, desenvolvido pela Mathwave em 2004, é um analisador de dados e aplicação de simulação que permite ajustar vários dados a diferentes distribuições de probabilidade (tabela 2) e seleccionar de forma automática a distribuição que melhor se ajusta aos dados Para tal recorre a três testes, Kolmogorov-Smirnov, Anderson-Darling e Chi-Squared, sendo neste trabalho usado o primeiro. O EasyFit pode ser usado

autonomamente em ambiente Windows ou com o Microsoft Excel. Paralelamente fornece resultados sob a forma de diferentes tipos de gráficos, alguns valores estatísticos como média e desvio padrão e os principais parâmetros das distribuições criadas.

Após obtenção dos dados através do Tshark, estes são passados para o EasyFit na forma de vectores de dados, que são analisados construindo-se então a função de densidade de probabilidade (FDP) que melhor se adequa aos dados. Finalmente, esta função é apresentada na forma gráfica.

Bernoulli	*Logistic*	*Nakagami*
Beta	*Generalized Pareto*	*Normal*
Binomial	*Geometric*	*Pareto*
Burr	*Gumbel Max*	*Pareto 2 (Lomax)*
Cauchy	*Gumbel Min*	*Pearson 5*
Chi-Squared	*Hyperbolic Secant*	*Pearson 6*
Dagum	*Hypergeometric*	*Pert*
Discrete Uniform	*Inverse Gaussian*	*Poisson*
Erlang	*Johson SB*	*Phased Bi-Exponential*
Error	*Johnson SU*	
Error Function	*Kumaraswamy*	*Phased Bi-Weibull*
Exponential	*Laplace*	*Power Function*
F	*Levy*	*Rayleigh*
Fatigue Life	*Logarithmic*	*Reciprocal*
Frechet	*Logistic*	*Rice*
Gamma	*Log-Gamma*	*Student's t*
Generalized	*Log-Logistic*	*Triangular*
Extreme Value	*Log-Pearson 3*	*Uniform*
Generalized	*(LP3)*	*Wakeby*
Gamma	*Lognormal*	*Weibull*
Generalized	*Negative Binomial*	

Tabela 2 – Distribuições suportadas pelo EasyFit

3.3 – Matlab

MATLAB (MATrix LABoratoy) [50] é criado nos finais da década de 70 do século passado por Cleve Moler que no ano de 1983, em conjunto com Jack Little e Steve Bangert, rescreve o MATLAB em linguagem C e funda a MathWorks.

Trata-se de um software interactivo de alto desempenho destinado ao cálculo numérico. Integra cálculo com matrizes, processamento de sinais, processamento de imagens, construção de gráficos, entre outros. O seu ambiente de trabalho é de fácil utilização e compreensão. A forma como o MATLAB trata as matrizes permite a resolução de vários problemas numéricos num tempo inferior ao que se gastaria ao escrever um programa semelhante noutros tipos de linguagem como C ou Fortran.

MATLAB foi adoptado pela primeira vez por engenheiros de projecto de controlo e rapidamente se espalhou para outros campos de aplicação, sendo agora utilizado nas áreas da educação e muito popular entre os cientistas das mais diversas áreas tais como processamento de imagem e controlo.

No âmbito deste trabalho o MATLAB é usado, partindo dos dados obtidos pelo Tshark, para cálculos estatísticos como obtenção de médias e geração de gráficos.

4 - Obtenção de dados

O principal objectivo deste trabalho, como foi anteriormente explicado, é o estudo de técnicas de detecção de intrusões para detecção de *port scans* recorrendo apenas a dados estatísticos do tráfego na rede. De seguida, serão apresentados exemplos dos dados recolhidos, bem como dos processos utilizados na sua obtenção, de forma a tornar possível a detecção *de port scans* na rede nos vários cenários pré-existentes.

Os diferentes dados são obtidos a partir de doze capturas de tráfego, efectuadas no servidor e com duração de 30 minutos cada, correspondentes a cada um dos tipos anteriormente referidos.

Para tal cada captura é passada pelo Tshark usando diferentes sintaxes (apresentadas em anexo) consoante os dados que se pretendem obter, sendo os resultados posteriormente passados para o EasyFit e/ou Matlab para posterior análise.

4.1 – Número de *flags* SYN por segundo

Para a obtenção do número de *flags* SYN isoladas do protocolo TCP (só o segundo LSB bit do octeto treze do protocolo TCP activo) ou não isoladas (segundo LSB bit do octeto treze do protocolo TCP activo e outros bits do mesmo octeto também activos), é usada a seguinte expressão composta:

Tshark –r infile.pcap –q –z io,stat,interval,"[filter],[filter]" > outfile.txt

A opção *–r* dá o parâmetro *infile.pcap* como parâmetro de entrada para ser lido. A opção *–q* é usada para que apenas sejam mostradas as estatísticas recolhidas pela expressão *–z io, stat, interval*. O campo *interval* é o intervalo temporal de amostragem, que neste caso tem o valor 1 (um segundo).

O campo *filter* pode ser omitido, usado apenas uma vez ou ser usado várias vezes. Para a obtenção do número de *flags* SYN o campo *filter* toma o valor:

COUNT(tcp)tcp[13]&2, COUNT(tcp)tcp[13]==2

Na figura 15 é possível ver o ficheiro de saída obtido com o comando apresentado. Na primeira coluna é dado o intervalo de tempo, na segunda a contagem das *flags* SYN não isoladas e por fim, na terceira coluna a cotagem do número de *flags* SYN isoladas.

```
================================================================
IO Statistics
Interval: 1.000 secs
Column #0: COUNT(tcp)tcp[13]&2
Time                |           COUNT |           COUNT
000.000-001.000                     2                   1
001.000-002.000                     4                   2
002.000-003.000                     6                   3
003.000-004.000                     4                   2
004.000-005.000                    10                   5
005.000-006.000                     6                   3
006.000-007.000                     8                   4
007.000-008.000                     8                   4
008.000-009.000                     6                   3
009.000-010.000                     2                   1
010.000-011.000                     8                   4
011.000-012.000                     4                   2
012.000-013.000                     8                   4
013.000-014.000                     4                   2
014.000-015.000                     4                   2
015.000-016.000                     0                   0
016.000-017.000                     0                   0
017.000-018.000                    10                   5
018.000-019.000                    14                   7
019.000-020.000                     8                   4
020.000-021.000                     2                   1
021.000-022.000                    12                   6
022.000-023.000                     2                   1
023.000-024.000                    10                   5
024.000-025.000                     2                   1
025.000-026.000                     6                   3
026.000-027.000                    14                   7
027.000-028.000                     8                   4
```

Figura 15 – Exemplo de ficheiro de saída do Tshark para o número de *flags* SYN

4.2 – Intervalo de tempo entre pacotes

Frame time delta corresponde ao intervalo de tempo entre a recepção de cada pacote. Este tempo é contabilizado pelo Wireshark aquando a captura e através do Tshark é possível visualizar o seu valor. Para tal é usada a seguinte sintaxe:

tshark -r infile.pcap -T fields –e[field] > outfile.txt

A opção *–T* configura o formato de saída aquando da descodificação dos pacotes da captura especificada pelo campo *–r* no nosso caso a opção usada é *fields* que indica que a saída a guardar no ficheiro *outfile.txt* deverá ser aquela especificada pelo campo *field* da opção *–e*. O campo *field* toma o valor:

frame.time_delta

Um exemplo do ficheiro de saída pode ser visto na figura 16, que apresenta o intervalo entre a chegada de dois pacotes consecutivos, considerando que o primeiro chega no instante zero.

```
0.000000000
0.000500000
0.000005000
0.024998000
0.000005000
0.000503000
0.000004000
0.000557000
0.000004000
0.000539000
0.000004000
0.012587000
0.017218000
0.000006000
0.008878000
0.000004000
0.000500000
0.000004000
0.002589000
0.000005000
0.002604000
0.000004000
0.004434000
0.000004000
```

Figura 16 – Exemplo de um ficheiro de saída do Tshark para o intervalo entre chegadas de pacotes

4.3 – Portos TCP quando só as *flags* SYN se encontram activas

O protocolo TCP usa o conceito de porto para identificar o envio e a recepção de aplicações/serviços. Cada lado da ligação TCP tem associado um porto de 16 bits (que varia entre 0 – 65535). Alguns serviços são tipicamente acessíveis em portos fixos, conhecidos como *Well-know port numbers* (portos bem conhecidos), que variam entre 1 e 1023, e são atribuídos pela IANA. Existem outras duas gamas de portos, portos registados (variam entre 1024 e 49151) e portos dinâmicos ou privados (variam entre 49152 e 63535) [51].

Cada pacote TCP apresenta o seu porto de origem e destino, o porto de origem (*source port*) ocupa os bits +0 até +7 dos octetos +0 e +1 no cabeçalho TCP e o porto de destino (*destination port*) ocupa os bits +0 até +7 dos octetos +2 e +3 do cabeçalho TCP.

O porto de origem é atribuído ao cliente e usado para acompanhar a sua sessão.

O porto de destino é usado para distribuir os pacotes num servidor para a aplicação de rede apropriada. Por exemplo, o porto 80 é o porto para tráfego HTTP e pacotes com destino ao porto 80 são processados por um servidor Web.

No nosso caso, apenas são recolhidos os portos de origem e destino dos pacotes TCP que contêm também só a *flag* SYN activa, ou seja, apenas os portos origem e destino aquando do pedido de início de sessão TCP.

Para a obtenção destes dados a partir da filtragem do Tshark, é usada a expressão:

Tshark –r infile.pcap –n –z io,stat,interval,"[filter],[filter]" > outfile.txt

A opção *–r* indica o parâmetro *infile.pcap* como parâmetro de entrada a ser lido. Como o Tshark, por omissão, apresenta os portos mais comuns, não na forma numérica, mas através do seu nome, é usado a opção *–n* para desactivar a resolução destes nomes e permitir que sejam apresentados os seus valores numéricos, para posterior análise. A expressão *–z io,stat, interval* activa o modo estatístico do Tshark de forma a obter os dados pedidos nos campos *filter*. O campo *interval* toma o valor 0, de forma a ser possível obter todos os pacotes que cumpram as condições do campo *filter* em vez de uma estatística por tempo, tal como no caso apresentado no ponto 4.1.

O filtro, campo *filter*, usado para a obtenção destes dados é o seguinte:

tcp[13]==2

Com este filtro obtém-se uma listagem de todos pacotes TCP que possuem apenas a *flag* SYN activa, sendo que, devido à ausência da opção *–q*, são apresentados vários dados acerca de cada um dos pacotes filtrados, como se pode ver na figura 17. Entre os dados apresentados estão os campos porto de origem e porto de destino, respectivamente na sétima e nona coluna,

Os resultados são guardados no ficheiro outfile.txt.

De forma a tornar a análise mais simples os valores do porto de origem e destino são retirados para um novo ficheiro fazendo uso do awk [52], uma linguagem de linha de comandos, usada em Linux, para a manipulação de *strings* e arquivos de texto. A expressão usada é a seguinte:

awk '{print $n;} inflile.txt > outfile.txt

A coluna número *n,* que toma o valor 7 ou 9*,* do ficheiro de entrada infile.txt é guardada no ficheiro de saída outfile.txt. Os ficheiros de entrada são os obtidos anteriormente, tal como aquele que é exemplificado na figura 17. Na figura 18 podemos ver exemplos dos ficheiros de saída contendo os portos de origem e destino filtrados.

```
590   0.983822  192.168.4.2 -> 192.168.1.2  TCP 33334 > 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197899031 TSER=0 WS=6
692   1.095332  192.168.8.6 -> 192.168.1.2  TCP 54888 > 21415 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197899049 TSER=0 WS=6
1176   1.891712 192.168.17.3 -> 192.168.1.2  TCP 60256 > 25 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=193470341 TSER=0 WS=6
1295   2.155597 192.168.12.3 -> 192.168.1.2  TCP 35420 > 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=193470406 TSER=0 WS=6
1542   2.421895 192.168.26.6 -> 192.168.1.2  TCP 44735 > 21 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197223560 TSER=0 WS=6
1635   2.588275 192.168.24.2 -> 192.168.1.2  TCP 40557 > 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197223593 TSER=0 WS=6
2537   3.924437 192.168.17.4 -> 192.168.1.2  TCP 48963 > 21 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=193470849 TSER=0 WS=6
2577   3.972554 192.168.24.5 -> 192.168.1.2  TCP 47982 > 47278 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197223940 TSER=0 WS=6
2666   4.095156 192.168.14.6 -> 192.168.1.2  TCP 54047 > 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=193470891 TSER=0 WS=6
2881   4.277294 192.168.26.3 -> 192.168.1.2  TCP 53573 > 41185 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197224024 TSER=0 WS=6
3172   4.503308  192.168.6.3 -> 192.168.1.2  TCP 54741 > 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197899858 TSER=0 WS=6
3401   4.680838  192.168.8.5 -> 192.168.1.2  TCP 36126 > 2324 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197899951 TSER=0 WS=6
3784   4.876856  192.168.6.2 -> 192.168.1.2  TCP 35238 > 1048 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197899992 TSER=0 WS=6
4891   5.453229 192.168.24.2 -> 192.168.1.2  TCP 45732 > 35882 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197224316 TSER=0 WS=6
5137   5.572555  192.168.6.4 -> 192.168.1.2  TCP 53428 > 54430 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197900172 TSER=0 WS=6
5770   5.952727 192.168.17.2 -> 192.168.1.2  TCP 42453 > 21 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=193471254 TSER=0 WS=6
5863   6.002906 192.168.14.4 -> 192.168.1.2  TCP 58364 > 60038 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=193471367 TSER=0 WS=6
6310   6.335612 192.168.14.6 -> 192.168.1.2  TCP 52180 > 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=193471451 TSER=0 WS=6
6495   6.503124 192.168.12.2 -> 192.168.1.2  TCP 42578 > 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=193471491 TSER=0 WS=6
6576   6.573428 192.168.22.4 -> 192.168.1.2  TCP 56354 > 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197224598 TSER=0 WS=6
7416   7.191433 192.168.17.3 -> 192.168.1.2  TCP 60999 > 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=193471665 TSER=0 WS=6
7533   7.297554  192.168.8.3 -> 192.168.1.2  TCP 48019 > 61330 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197900608 TSER=0 WS=6
7796   7.566277 192.168.22.3 -> 192.168.1.2  TCP 59758 > 57063 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=197224846 TSER=0 WS=6
7829   7.604471 192.168.12.4 -> 192.168.1.2  TCP 58715 > 21 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=193471768 TSER=0 WS=6
```

Figura 17 – Exemplo de um ficheiro de saída do Tshark para os dados dos pacotes só com a *flag* SYN activa.

Portos de Origem	Portos de Destino
36758	41906
50390	21
52399	80
43844	17808
36743	21
51826	80
35495	21
52115	21
33419	80
53641	21
58223	80
52633	12123
50657	6125
46439	21
52877	24928
32904	21

Figura 18 – Exemplo dos ficheiros de saída com os portos de origem e destino

5 – Resultados

Após obtenção dos dados estatísticos das capturas estes são passados para o EasyFit e/ou MatLab para a obtenção das curvas de densidade de probabilidade correspondentes ou de outros dados estatísticos relevantes. Os resultados são apresentados por ambiente ou seja, para o caso de tráfego c*lean* e tráfego com as quatro topologias de *port scan* apresentadas na tabela 1, e, dentro de cada ambiente, para os diversos tipos de dados recolhidos

5.1 – Ambiente *Clean*

Os resultados obtidos através da análise do ambiente *clean* são o ponto de partida para a detecção de ataques. É a partir destes resultados que se irá estabelecer um padrão de comportamento tido como normal para posterior comparação com resultados que apresentem desvios relativamente a este padrão.

5.1.1 - Número de *flags* SYN por segundo

O número de ***flags* SYN isoladas** e o número de ***flags* SYN não isoladas** são melhor ajustadas por uma distribuição do tipo *Johnson-SB*, que pode ser consultada no anexo C1

Nas figuras 19 e 20 podem-se ver gráficos da função densidade de probabilidade (FDP) de um dos ficheiros de dados contendo o número de *flags* SYN isoladas e de *flags* SYN não isoladas por segundo, respectivamente, correspondentes ao ambiente *clean*.

Nos resultados das *flags* SYN isoladas verifica-se, para todos os casos analisados, que o valor mais comum de *flags* SYN por segundo (a moda) é de três, não existindo valores superiores a doze, e que o valor de pico da curva corresponde a uma probabilidade aproximada de 0,22, para uma construção gráfica com o número de colunas do histograma em modo automático.

Os resultados das *flags* SYN não isoladas é semelhante aos resultados das *flags* SYN isoladas, tendo como diferença assinalável o facto de ter maior número de *flags* SYN contabilizadas por segundo (aproximadamente o dobro).

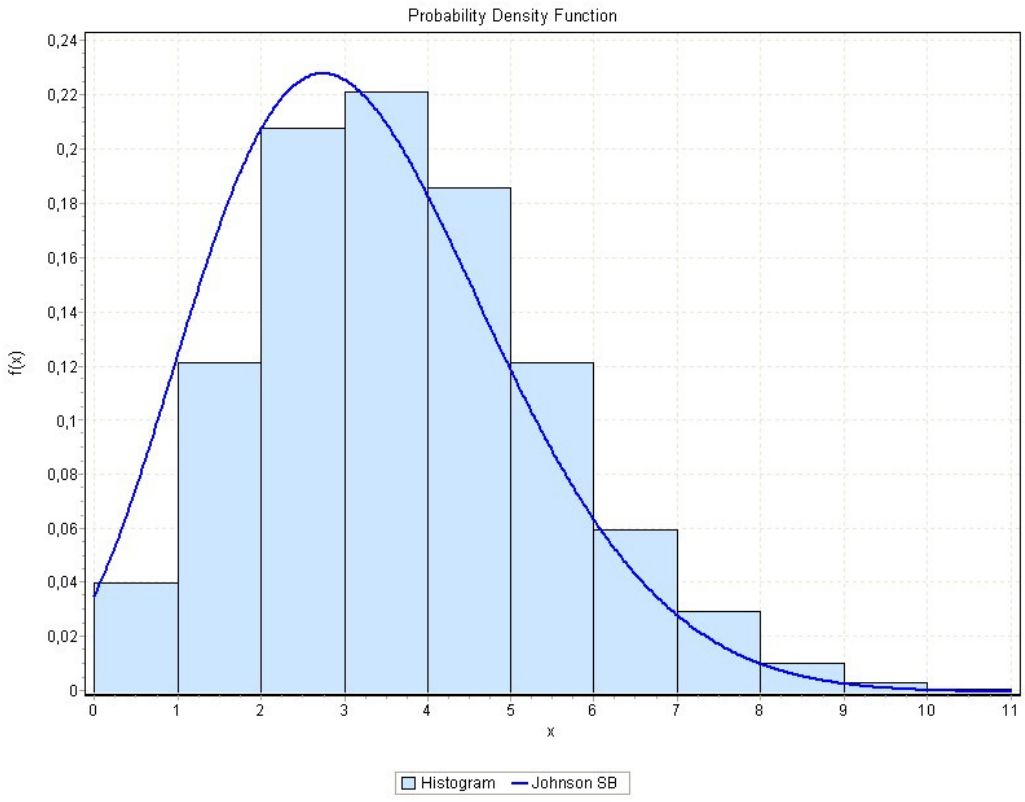

Figura 19 – FDP para *flags* SYN isoladas no ambiente *Clean*

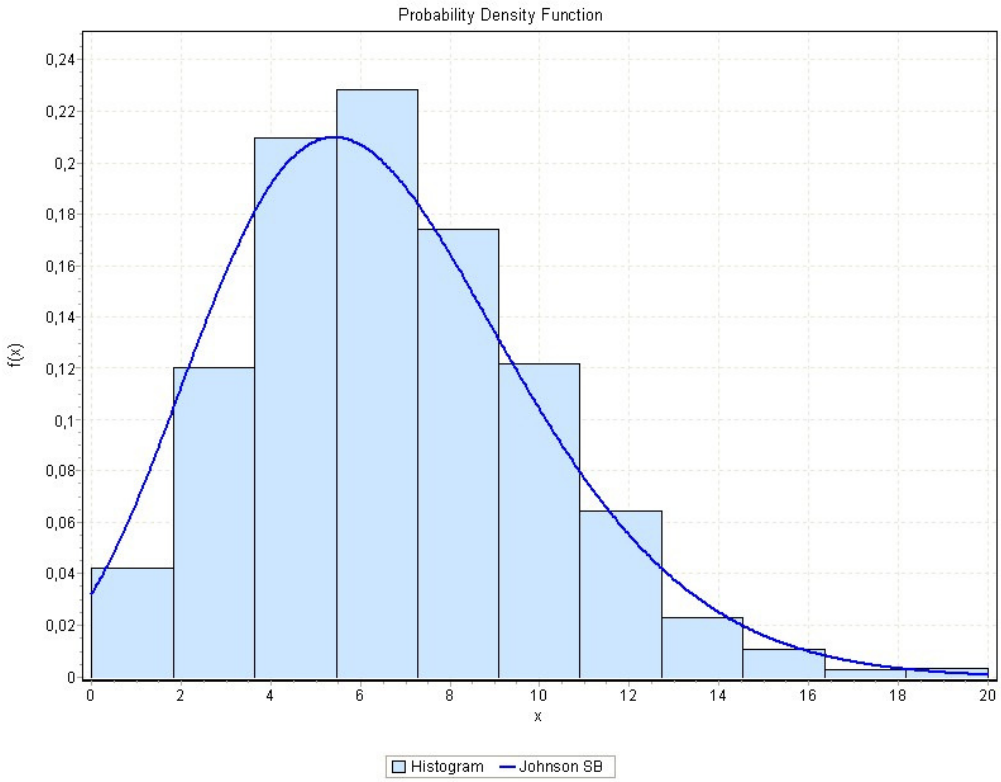

Figura 20 – FDP para *flags* SYN não isoladas no ambiente *Clean*

Recorrendo ao MatLab analisou-se a razão entre o número total de *flags* SYN não isoladas e o número total de *flags* SYN isoladas para cada um dos doze ficheiros de dados. Na tabela 3 são apresentados esses valores.

Número total de *flags* SYN não isoladas.	**Número total de *flags* SYN isoladas.**	**Razão entre número de *flags* SYN não isoladas e isoladas**
10007	5003	2,00019988007196
11351	5675	2,00017621145374
11593	5796	2,00017253278123
11399	5699	2,00017546938059
11560	5780	2

11556	5778	2
11586	5792	2,00034530386740
11410	5705	2
11516	5757	2,00034740316137
11428	5713	2,00035007876772
11502	5750	2,00034782608696
11622	5809	2,00068858667585
Média		
11377,5	5688,08333333333	2,00023440819256

Tabela 3 – Razão entre número de *flags* SYN não isoladas e isoladas no ambiente *clean*

De notar que nos doze casos analisados a razão entre o número total de *flags* SYN não isoladas e isoladas é sempre superior ou igual a 2,assim como a razão das médias.

5.1.2 – Intervalo de tempo entre pacotes

Devido à limitação do EasyFit apenas suportar vectores com um máximo de 250 000 linhas e tendo os ficheiros de dados, para este campo, mais de um milhão de entradas, apenas os primeiros 250 000 valores do vector de dados são passados para o EasyFit. Por uma questão de tempo de processamento, dos 250 000 valores, apenas os 50 000 valores no intervalo [150 000 – 200 000] são considerados.

A distribuição que melhor se adapta ao intervalo considerado é do tipo *Fatigue Life* (anexo C2)

É possível observar graficamente que a quase totalidade dos pacotes, para todos os casos analisados, chega com intervalos entre si inferiores a 0.005 segundos sendo a média do caso apresentado na

figura 21 igual a 0,00124 segundos. De notar a diferença entre a média gráfica e a média real obtida em MatLab. Graficamente, a maioria dos valores cai no intervalo entre 0 e 0,005, pois a construção do gráfico de densidade de probabilidade é feita com base num histograma que divide a gama de valores em 60 intervalos (barras), daí o facto de a grande maioria dos valores cair dentro do primeiro intervalo.

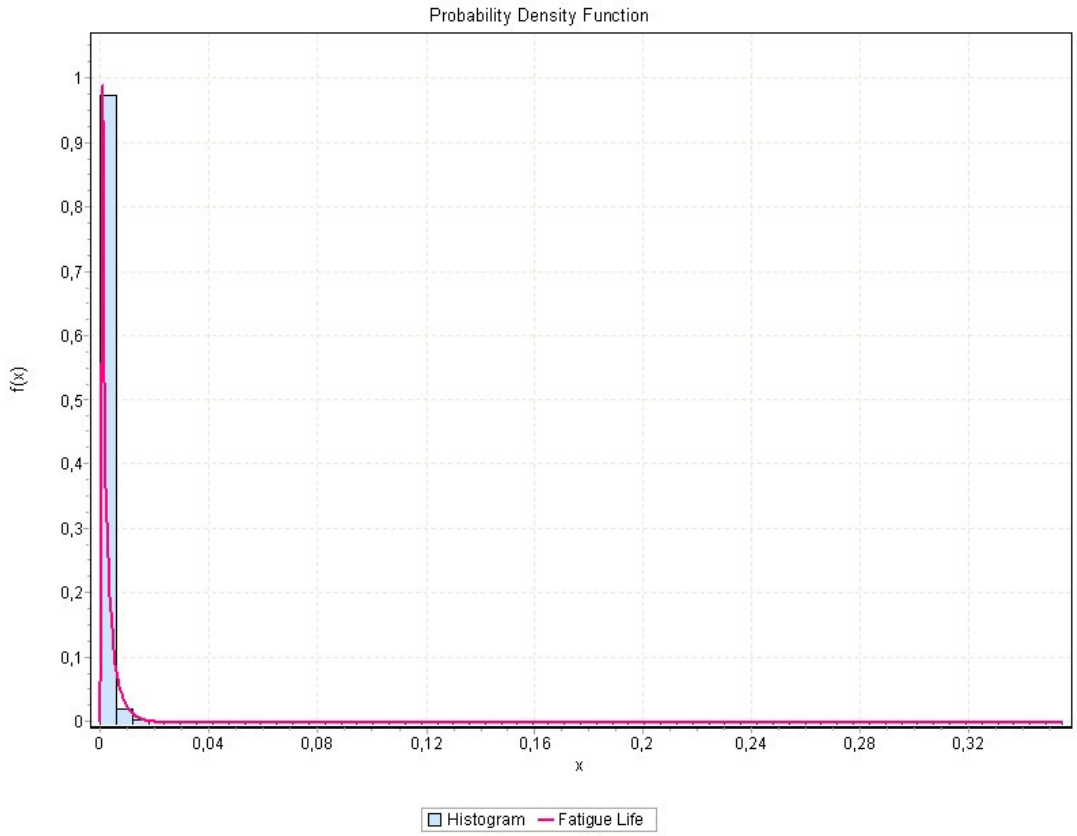

Figura 21 – FDP para o intervalo de tempo entre pacotes no ambiente *Clean*

5.1.3 – Portos TCP quando só as *flags* SYN se encontram activas

O *EasyFit* permite a construção de histogramas (para posterior desenho dos gráficos das funções de densidade de probabilidade) em

modo automático, atribuindo de forma automática o número de barras do histograma, ou em modo manual, no qual é possível especificar o número de barras do histograma, até um máximo de 200. Dado o elevado número de portos (mais de 63000), optou-se pelo modo manual e considerando o valor máximo permitido, para uma melhor leitura gráfica.

Os **portos de origem** no ambiente *clean* obedecem a uma distribuição do tipo *Wakeby*, descrita no anexo C3

Na figura 22 está ilustrado um exemplo de um dos gráficos obtidos para os portos de origem do ambiente *clean*.

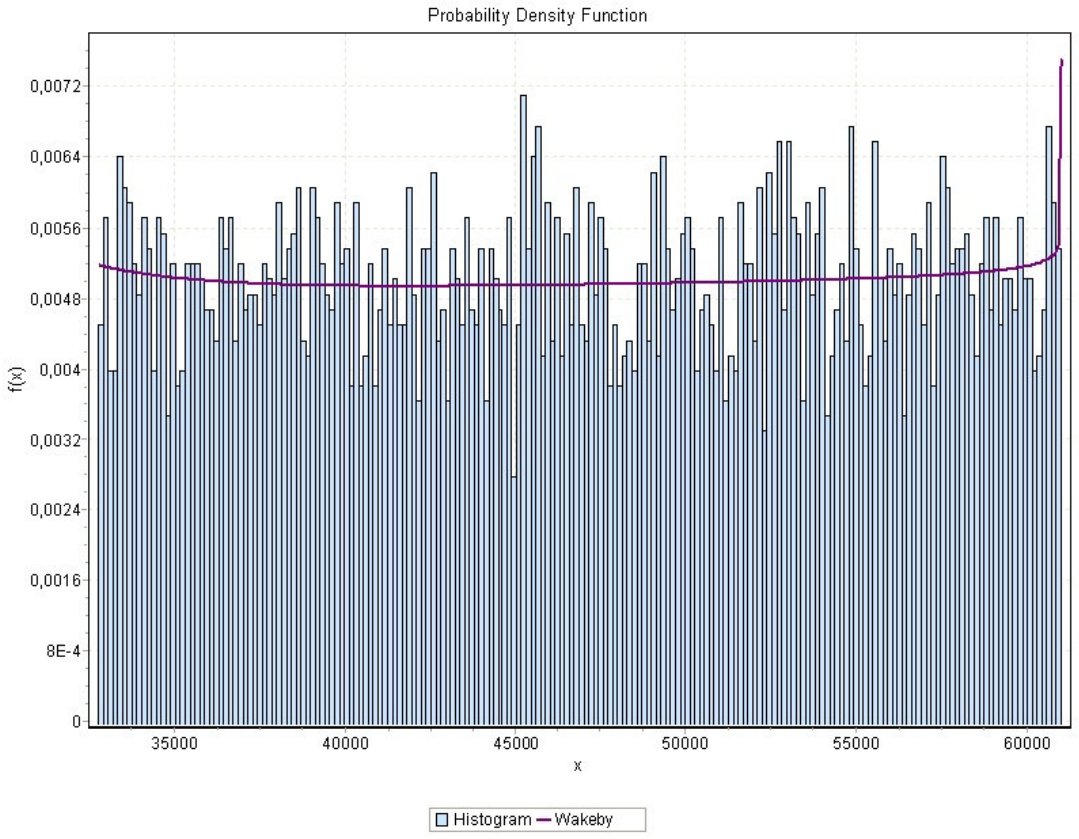

Figura 22 – FDP para os portos de origem no ambiente *Clean*

Como se pode ver através do exemplo da figura 22, os portos de origem são acedidos de forma mais ou menos uniforme, não

havendo grande prevalência de nenhum porto em relação aos restantes.

Através da análise em Matlab pode-se também concluir que os portos de origem têm, em todos os casos analisados, valores superiores a 32000.

Relativamente aos dados referentes ao **porto de destino** em ambiente *clean*, a distribuição que melhor se ajusta é a distribuição *Phased Bi-Exponential* descrita no anexo C4*.*

Na figura 23 é possível ver um dos gráficos obtidos para os portos de destino obtidos no ambiente *clean*.

Em todos os casos analisados observa-se que a maior parte dos pacotes (cerca de 70%) tem como porto de destino um valor que se enquadra dentro do primeiro intervalo do histograma (de 0 a aproximadamente 320). Tal deve-se ao facto de os portos mais comuns terem valores compreendidos entre 0 e 1023 e, destes, os mais comuns no tráfego que serve de base a este trabalho serem os portos 20 e 21, correspondentes a tráfego FTP, porto 25 correspondente a SMTP, porto 80 correspondente a HTTP, porto 110 correspondente a POP3 e portos 161 e 162 correspondentes a SNMP.

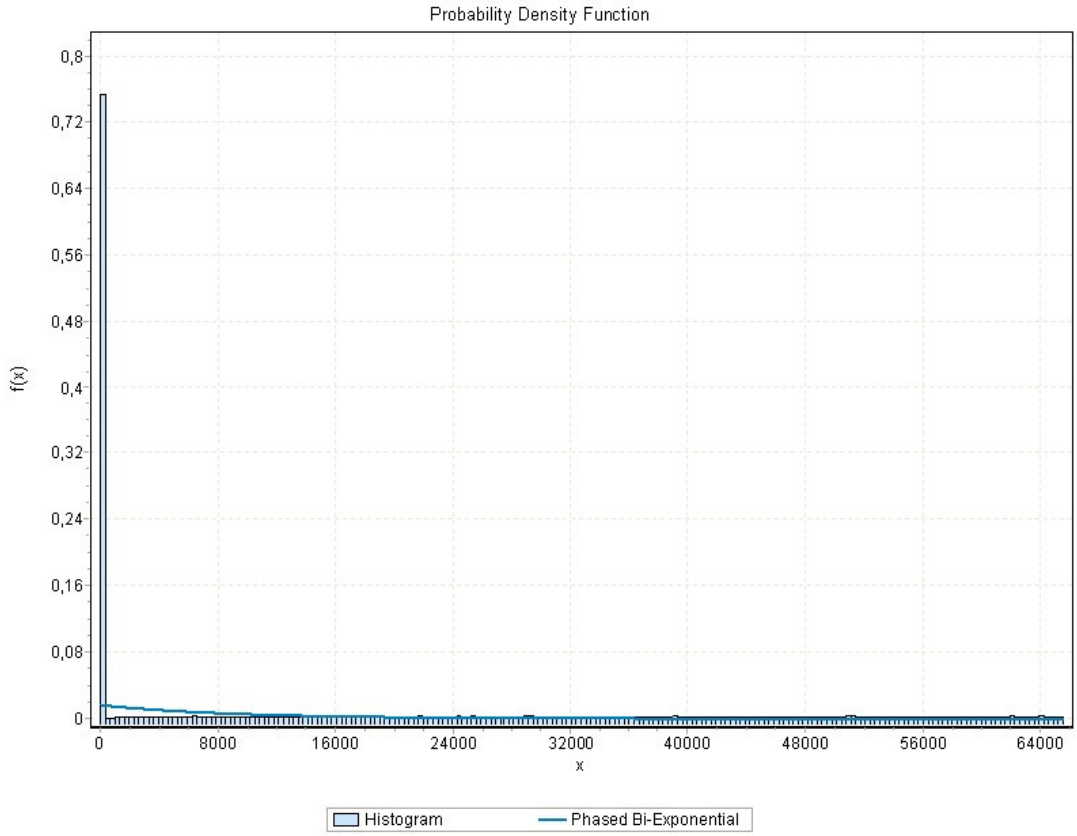

Figura 23 – FDP para os portos de destino no ambiente *Clean*

5.2 – Ambiente *Port scan* 1 para 1

5.2.1 - Número de *flags* SYN por segundo

Como referido anteriormente o tráfego gerado em ambiente de *port scan* 1 para 1 corresponde a de três modos distintos: *aggressive*, *normal* e *sneaky*.

Para os **modos *aggressive* e *normal* o número de *flags* SYN isoladas** é ajustado da melhor forma pela distribuição do tipo *Log-Logisitc*, descrita no anexo C5.

As figuras 24 e 25 representam os gráficos das funções densidade de probabilidade para dois dos casos analisados de *port scan* 1 para 1, no modo *aggressive* e *normal,* respectivamente.

Através da análise dos gráficos das funções de densidade de probabilidade dos casos *aggressive* e *normal* observa-se uma grande semelhança entre eles e uma grande diferença quando comparados com o ambiente *clean*.

Apesar da grande maioria das contagens de *flags* SYN por segundo em ambos os casos ser inferior a dez, existem algumas contagens com um elevado número de *flags* SYN isoladas, chegando a valores acima das 400 por segundo, ou seja, existem momentos onde o número de pedidos de início de sessão TCP dispara, o que representa um comportamento bastante distinto daquele que era apresentado pelo ambiente *clean*.

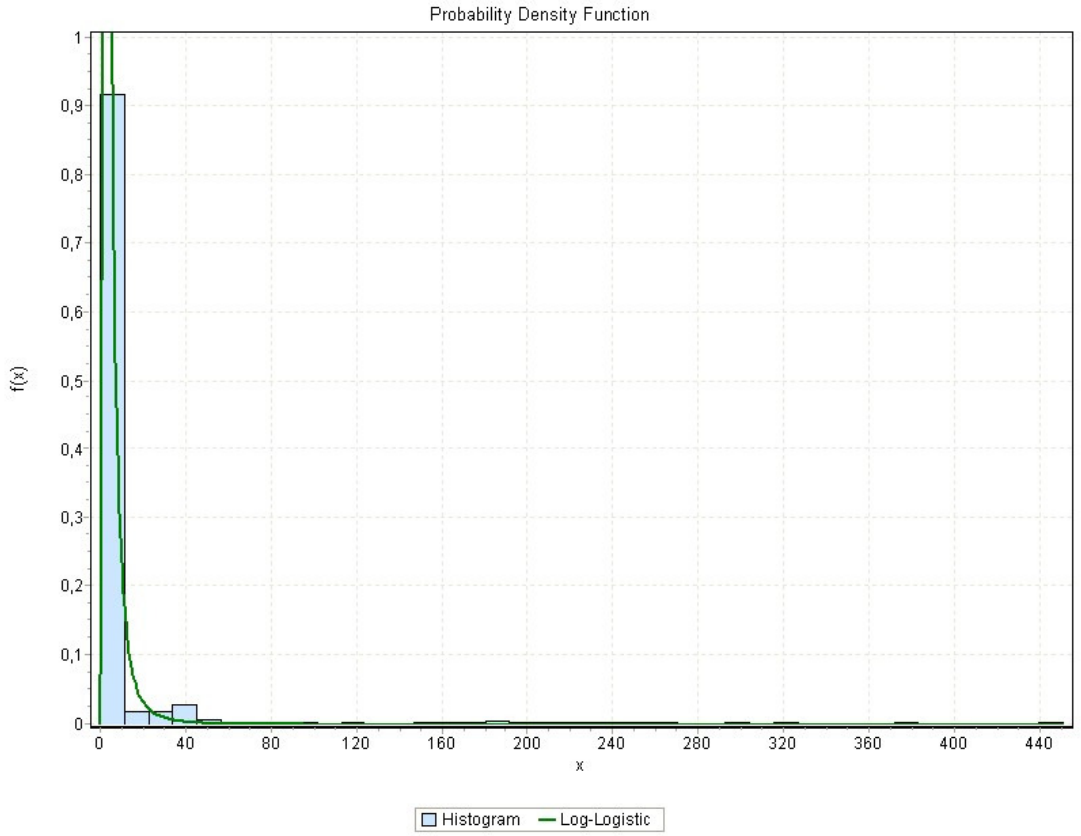

Figura 24 – FDP para *flags* SYN isoladas no ambiente *port scan* 1 para 1 *aggressive*

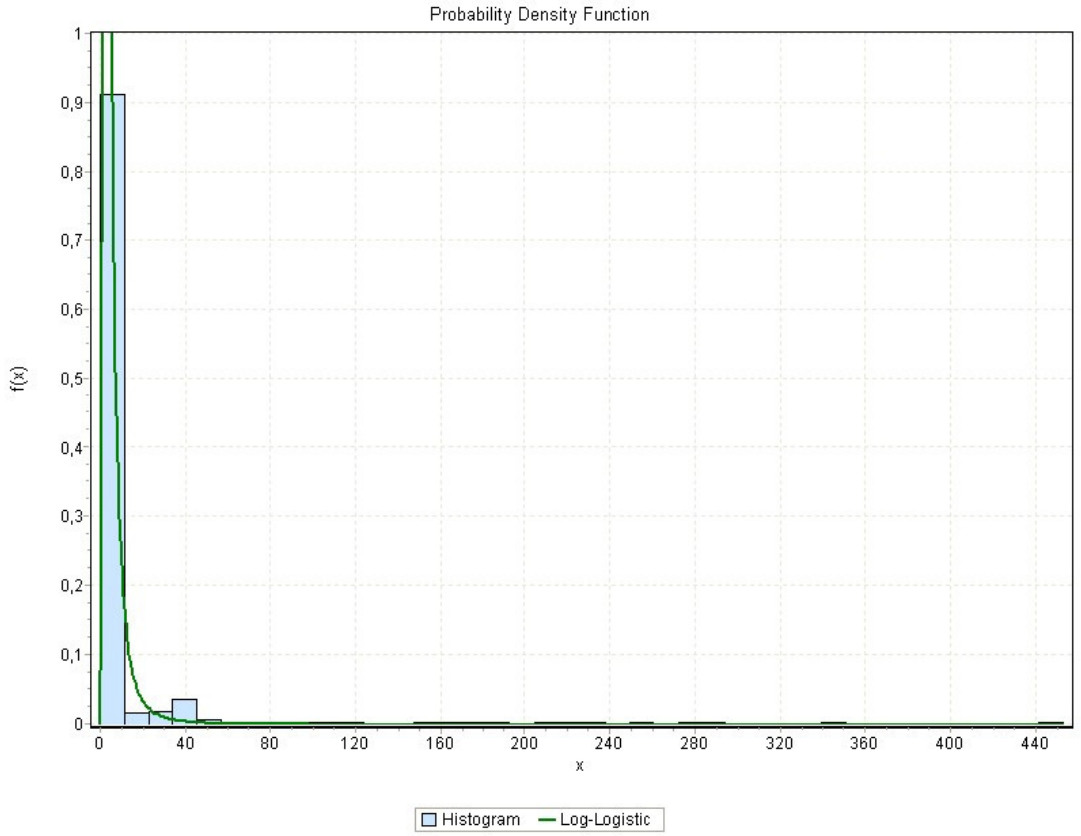

Figura 25 – FDP para *flags* SYN isoladas no ambiente *port scan* 1 para 1 *normal*

O **número de *flags* SYN isoladas, no modo *sneaky***, no ambiente de *port scan* um para um, difere dos modos *aggressive* e *normal*. Apresenta uma distribuição do tipo *Johnson-SB*, como se pode ver na figura 26, tal como acontecia no cenário correspondente ao ambiente *clean*.

O número máximo de *flags* SYN por segundo situa-se na casa das dez, não ultrapassando em nenhum dos casos as treze por segundo, sendo a média próxima das três *flags* por segundo (3,2987 para o exemplo mostrado).

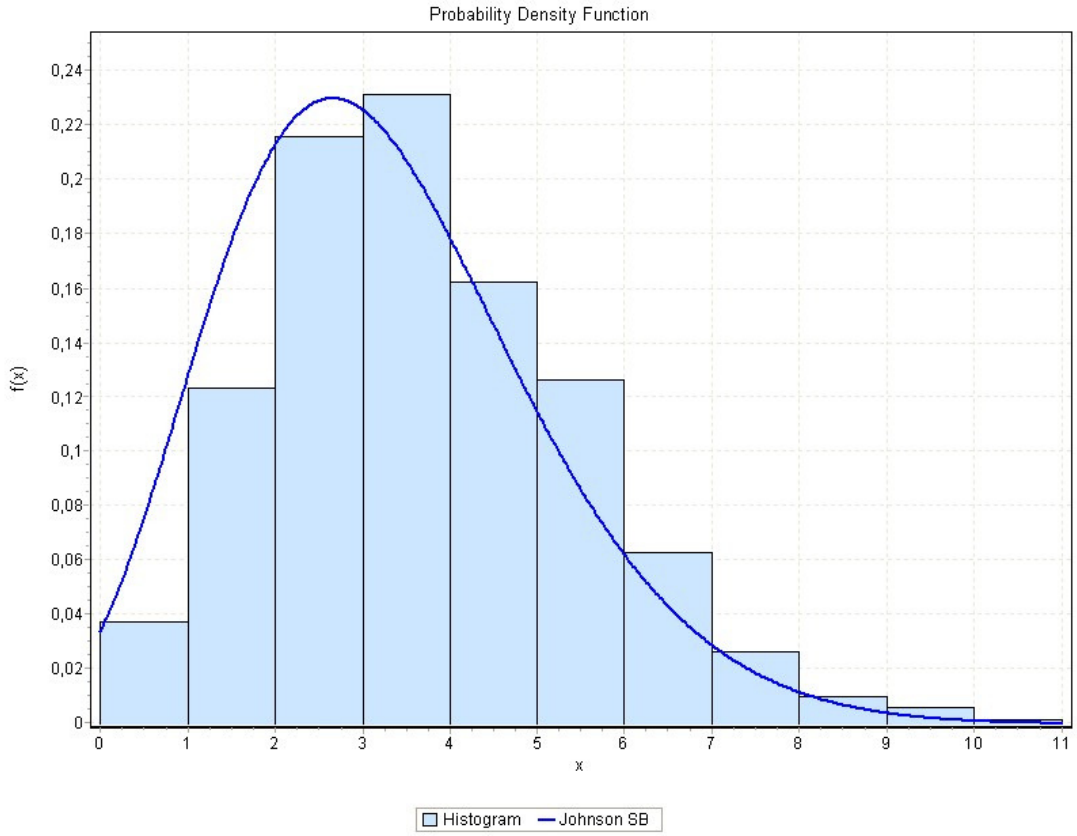

Figura 26 – FDP para *flags* SYN isoladas no ambiente *port scan* 1 para 1 *sneaky*

Relativamente ao número de ***flags* SYN não isoladas no modo *sneaky*** do ambiente *port scan* 1 para 1 podemos observar que também aqui a distribuição que melhor ajusta os valores obtidos é igual à obtida para o ambiente *clean*, ou seja, uma distribuição do tipo *Johson SB*, como se pode ver na figura 27, que mostra um dos gráficos obtidos.

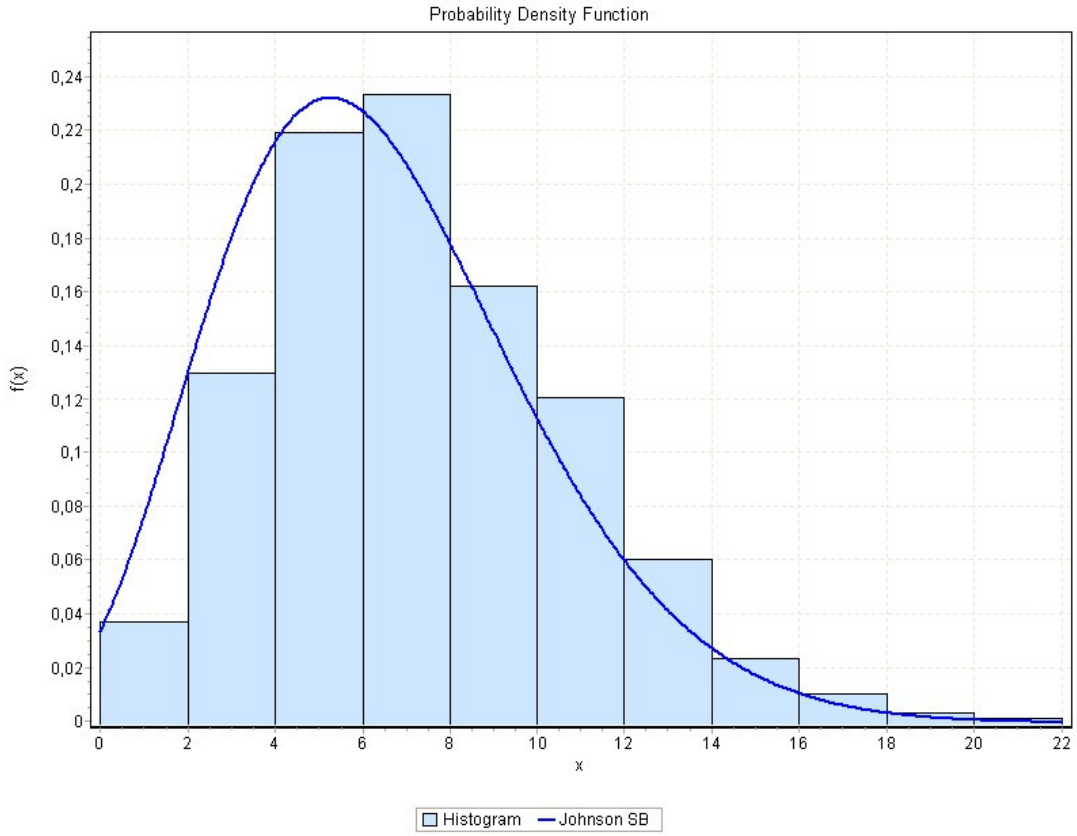

Figura 27 – FDP para *flags* SYN não isoladas no ambiente Port-Scan 1 para 1 modo *sneaky*

Uma vez mais se observa um baixo valor de *flags* SYN por segundo, não ultrapassando as 25 por segundo, sendo este valor sensivelmente o dobro do obtido para o caso das *flags* SYN isoladas.

O número de ***flags* SYN não isoladas por segundo nos modos *normal* e *aggressive*** apresentam ambos funções de densidade de probabilidade com distribuições do tipo *Log-Logistic*, tal como no o caso das *flags* SYN isoladas.

Nas figuras 28 e 29 são apresentados dois dos gráficos do número de *flags* SYN não isoladas obtidos respectivamente para o modo *normal* e *aggressive* no ambiente *port scan* 1 para 1.

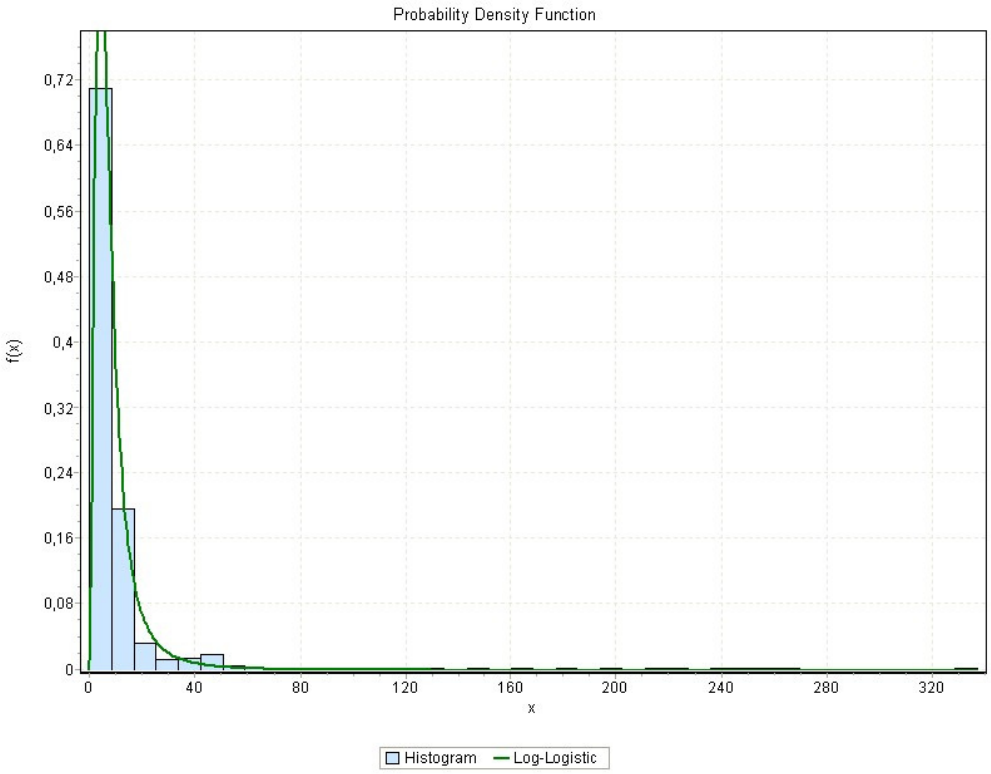

Figura 28 – FDP para *flags* SYN não isoladas no ambiente *Port scan* 1 para 1 modo *normal*

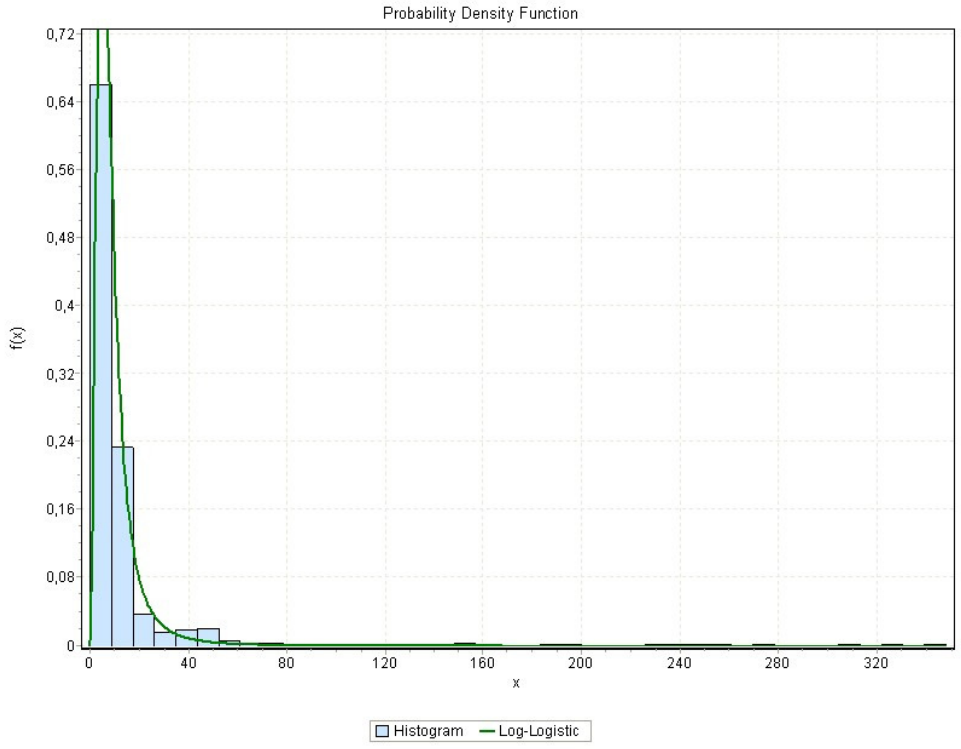

Figura 29 – FDP para *flags* SYN não isoladas no ambiente *Port scan* 1 para 1 modo *aggressive*

Na mesma linha do caso das *flags* SYN isoladas, verificam-se contagens de elevado valor no número de *flags* por segundo, apesar da maioria das contagens ser inferior a 20 *flags* por segundo. Esta situação torna o ambiente *port scan* 1 para 1, nos modos *normal* e *aggressive* distintos do ambiente *clean*.

Devido às semelhanças apresentadas pelo ambiente *clean* e pelo ambiente de *port scan* um para um no modo *senaky* os dados foram passados ao MatLab no sentido de comparar a razão entre o número total de *flags* SYN não isoladas e o número total de *flags* SYN isoladas. Na tabela 4 estão os valores obtidos.

Número total de *flags* SYN não isoladas.	**Número total de *flags* SYN isoladas.**	**Razão entre número de *flags* SYN não isoladas e isoladas**
10459	5269	1,98500664262668
11724	5916	1,98174442190669
11313	5711	1,98091402556470
11723	5915	1,98191039729501
11757	5932	1,98196223870533
11777	5941	1,98232620770914
11505	5807	1,98122955054245
11567	5837	1,98166866541031
11640	5874	1,98161389172625
11818	5931	1,99258135221716
11497	5802	1,98155808341951
11413	5760	1,98142361111111
Média		
11516,0833333333	5807,91666666667	1,982825166798185

Tabela 4 – Razão entre número de *flags* SYN não isoladas e isoladas para o ambiente *port scan* 1 para 1 modo *sneaky*.

Nos doze casos analisados a razão entre o número total de *flags* SYN não isoladas e isoladas nunca ultrapassa o valor de 2 e apenas num caso ultrapassa 1,98. Por comparação com os resultados do ambiente *clean,* pode ver-se que os valores médios para os números de *flags* SYN isoladas e não isoladas são superiores e que a razão entre as *flags* é inferior.

5.2.2 – Intervalo de tempo entre pacotes

A limitação imposta pelo EasyFit, descrita no ponto 5.1.2, também é aqui aplicada da mesma forma.

Os três modos existentes **no ambiente *port scan* 1 para 1, *sneaky***, ***normal*** e ***aggressive*,** apresentam a mesma função densidade de probabilidade para o intervalo de tempo entre pacotes, também igual à função correspondente ao ambiente *clean*. A distribuição é tipo *Fatique Life*, em que a quase totalidade dos pacotes chega com um intervalo de tempo entre si inferior a 0.005 segundos.

Na tabela 5 estão apresentadas as médias dos intervalos para os três casos apresentados nas figuras 30, 31 e 32 que representam os gráficos das funções de densidade de probabilidade para os modos *sneaky*, *normal* e *aggressive*, respectivamente.

	Média do intervalo de tempo entre pacotes
Sneaky	0,00112
Normal	0,00114
Aggressive	0,00123

Tabela 5 – Média do intervalo de tempo entre pacotes no ambiente *port scan* 1 para 1.

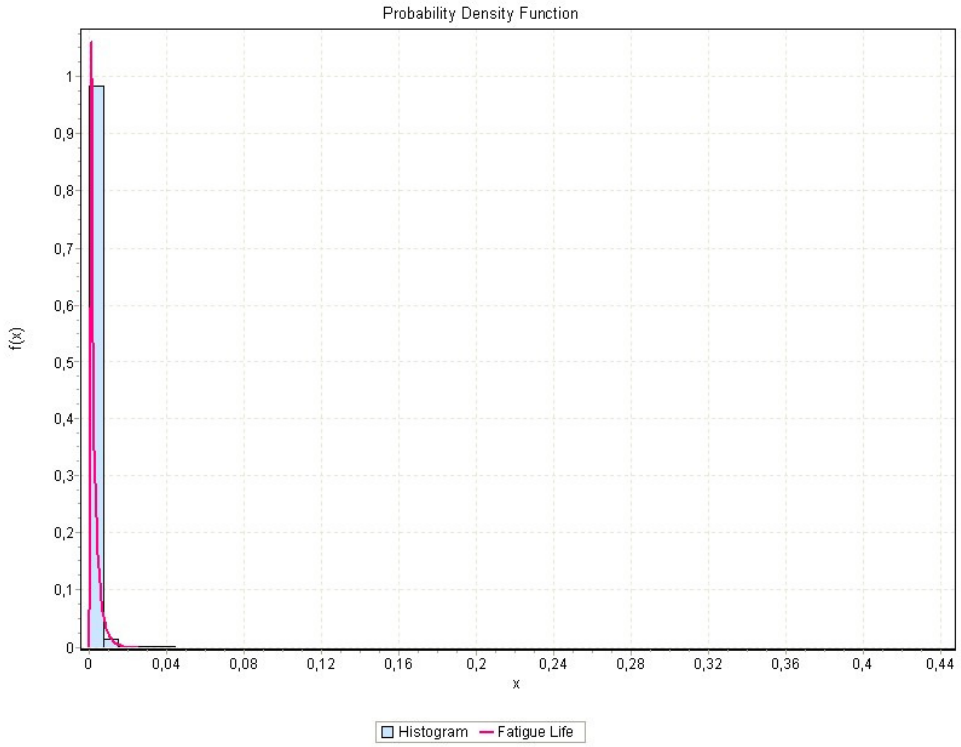

Figura 30 – FDP para o intervalo de tempo entre pacotes no ambiente *port scan* 1 para 1 modo *sneaky*

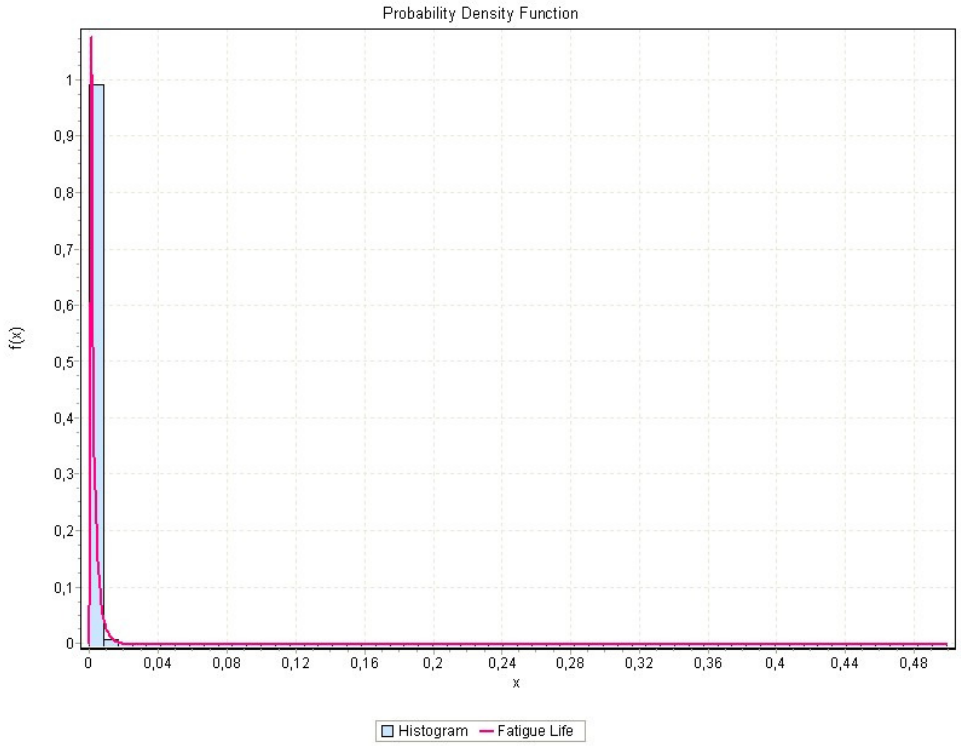

Figura 31 – FDP para o intervalo de tempo entre pacotes no ambiente *port scan* 1 para 1 modo *normal*

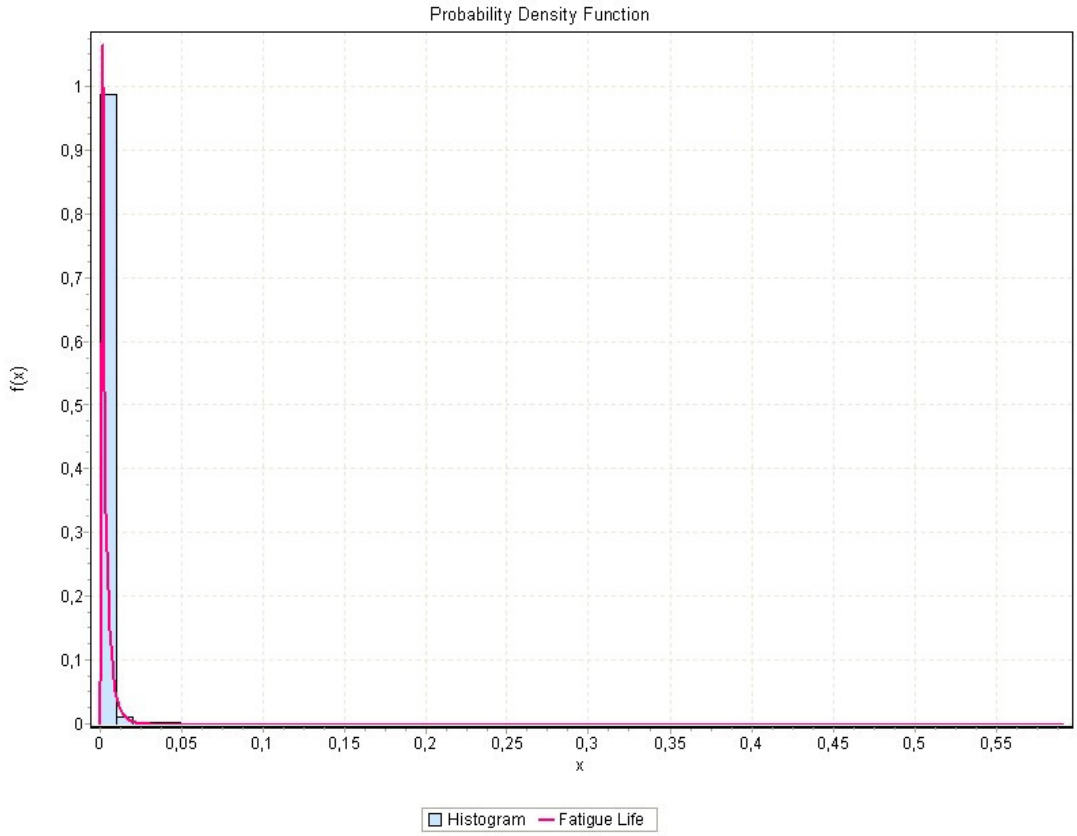

Figura 32 – FDP para o intervalo de tempo entre pacotes no ambiente *port scan* 1 para 1 modo *aggressive*

5.2.3 – Portos TCP quando só as *flags* SYN se encontram activas

Tal como no ambiente *clean*, também aqui os gráficos das funções de densidade de probabilidade apresentados são construídos com base em histogramas que dividem o intervalo dos dados em 200 intervalos de igual valor.

No **modo *aggressive* do ambiente *port scan* 1 para 1** a função de densidade de probabilidade dos portos de origem quando só as *flags* SYN se encontram activas são melhor ajustadas por uma distribuição do tipo *Wakeby* (anexo C3). Apesar da distribuição neste

caso ser a mesma do ambiente *clean* é possível observar diferenças entre ambos os ambientes. Na figura 33 está representado um dos gráficos obtidos.

Através da análise em MatLab, e tal como no ambiente *clean*, pode verificar-se que os portos de origem no ambiente *port scan* 1 para 1 modo *aggressive* apresentam valores acima dos 32000. Para todos os casos analisados, verifica-se a existência de picos em diferentes valores dentro da gama de portos de origem e acima do valor 32000.

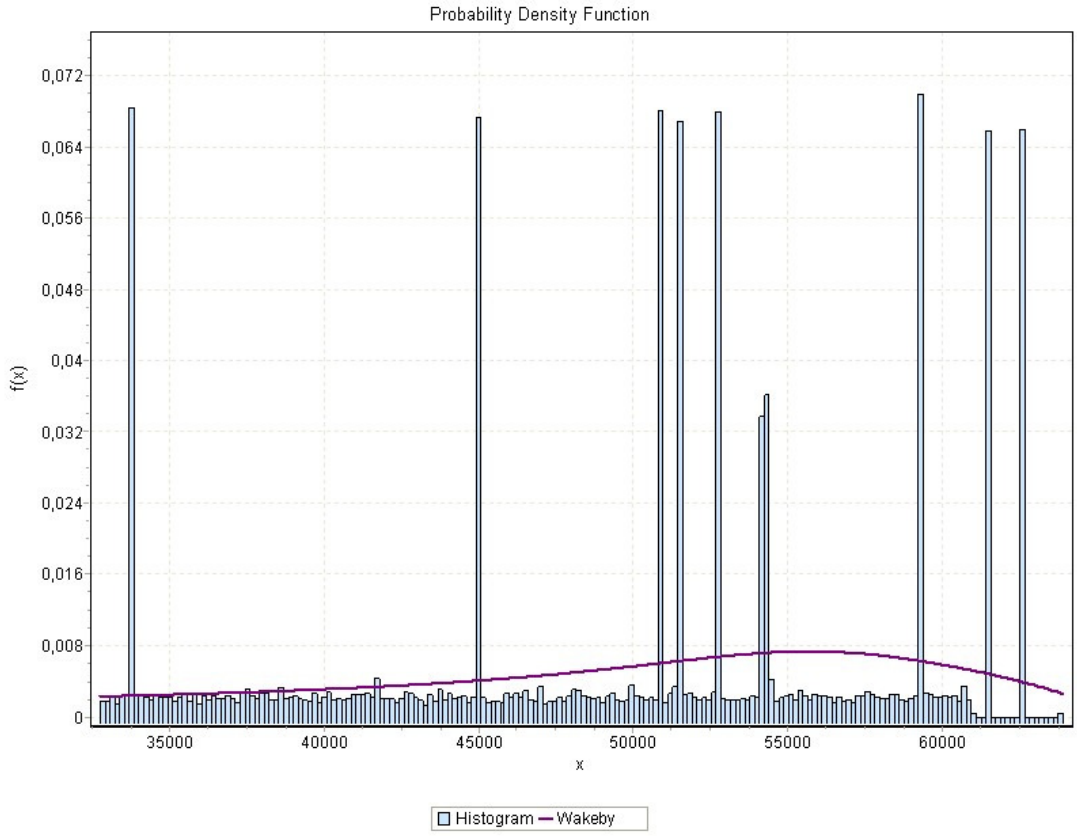

Figura 33 – FDP para os portos de origem no ambiente *port scan* 1 para 1 modo *aggressive*

O **modo *normal* do ambiente *port scan* 1 para 1** apresenta também uma distribuição do tipo *Wakeby* e, à imagem do modo *aggressive* e do ambiente *clean*, os portos de origem têm valores

acima do porto 32000, apresentando picos destacados em várias gamas de valores: verifica-se que a grande maioria dos intervalos corresponde a uma probabilidade aproximada de 0.002, enquanto que os picos apresentam valores 30 vezes superiores. Na figura 34 representa-se o gráfico da FDP de um dos casos estudados para este ponto.

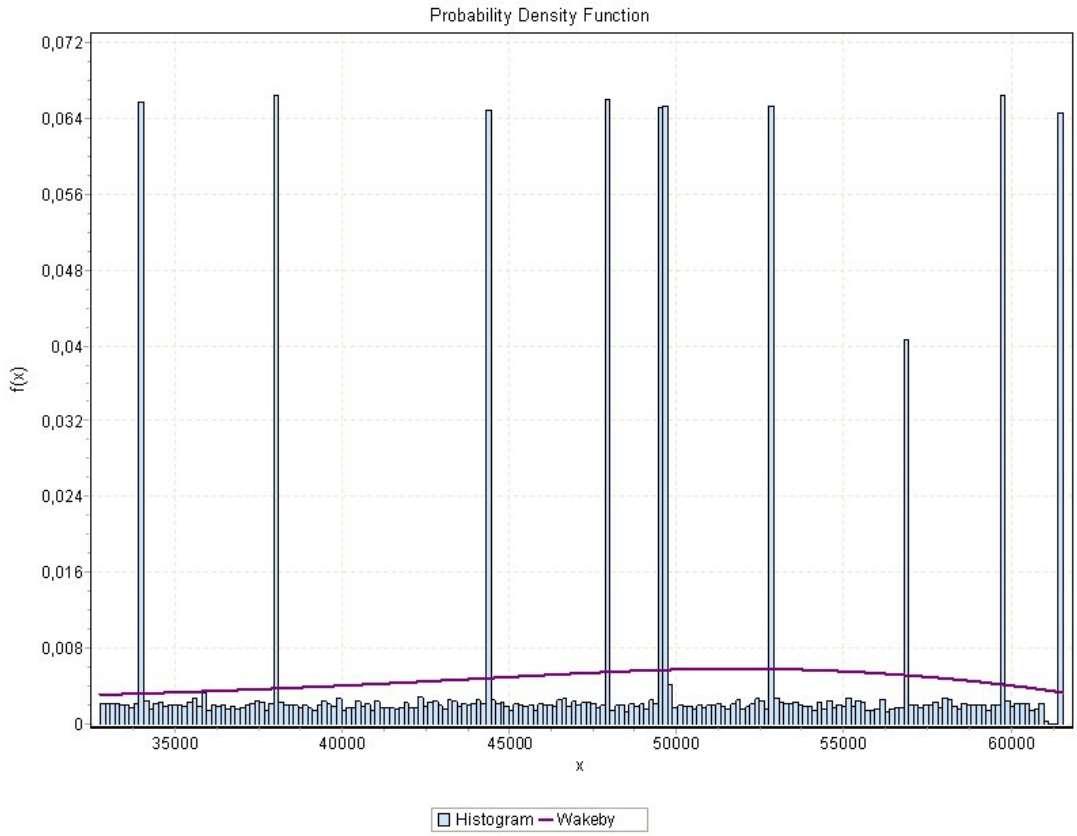

Figura 34 – FDP para os portos de origem no ambiente *port scan* 1 para 1 modo *normal*

Para o **modo *sneaky*, no ambiente *port scan* 1 para 1**, a FDP dos portos de origem quando só as *flags* SYN estão activas, cujo exemplo se encontra na figura 35, é ajustada por uma distribuição do tipo *Wakeby*. Apesar de apresentar uma distribuição igual ao ambiente *clean* e aos restantes modos deste ambiente, é possível observar algumas diferenças. Neste modo surge apenas um pico com um número de ocorrências do porto de origem cerca de 5 vezes

superior à média: este pico absorve os portos de origem com valores no intervalo 58250 a 58750.

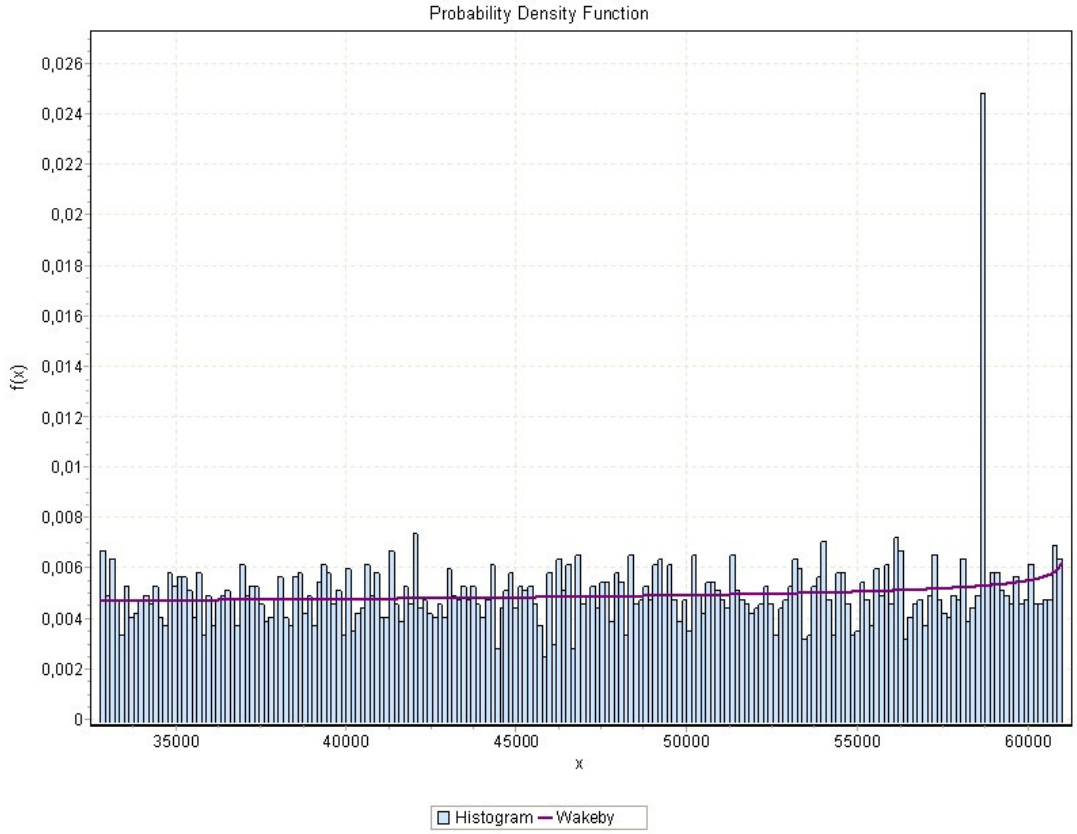

Figura 35 – FDP para os portos de origem no ambiente *port scan* 1 para 1 modo *sneaky*

Os **portos de destino** quando só as *flags* SYN estão activas para os modos *aggressive* e *normal* apresentam gráficos de densidade de probabilidade semelhantes, apesar de apresentarem distribuições distintas.

Para **o modo *aggressive* no ambiente *port scan* 1 para 1** a distribuição que melhor se adequa é do tipo *Phased Bi-Weibull* (anexo C6).

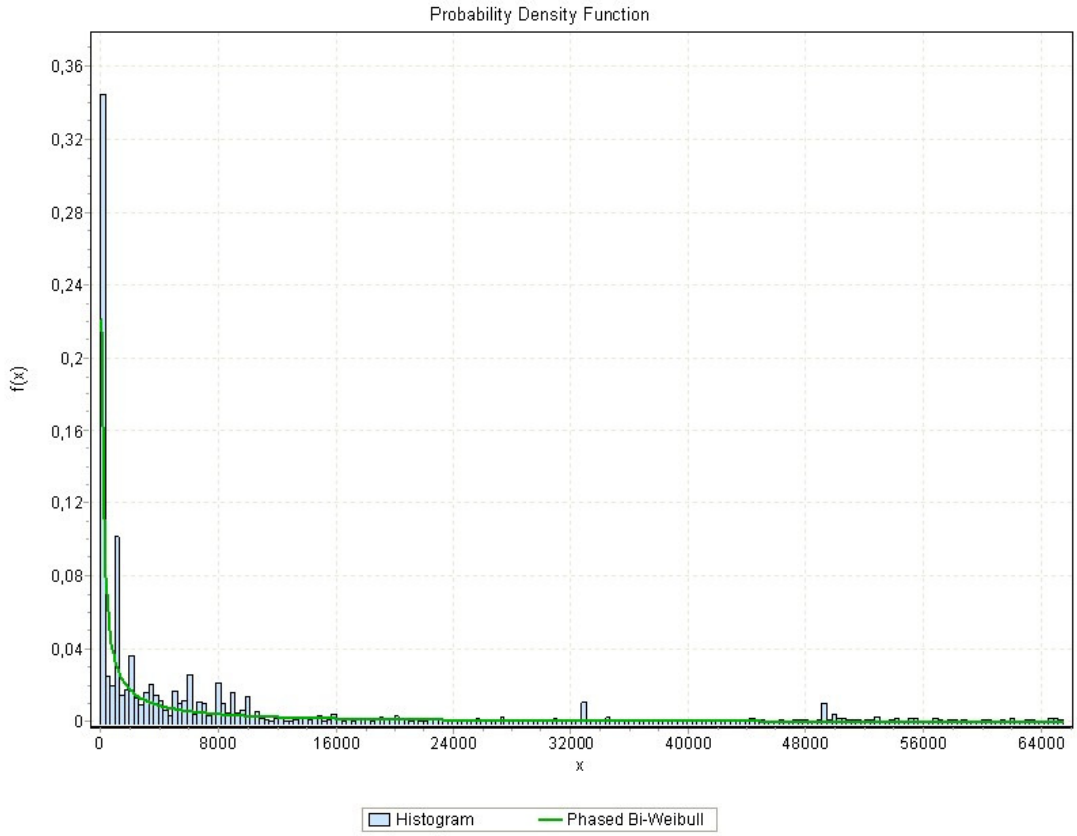

Figura 36 – FDP para os portos de destino no ambiente *port scan* 1 para 1 modo *aggressive*

Na figura 36 encontra-se um dos gráficos obtidos para a FDP. Da análise deste gráfico e dos restantes obtidos para este modo é possível observar que a maioria dos portos de destino tem valores baixos (inferiores a 10000), sendo que cerca de 35% dos portos de destino apresentam valores que se encaixam no primeiro intervalo do histograma. Tal pode ser explicado pelo facto já referido de os portos mais comuns terem valores entre 1 e 1023. Por comparação com o ambiente *clean,* observam-se diferentes tipos de função de densidade de probabilidade e apesar da maioria dos portos de destino se encontrar ainda no primeiro intervalo, como já referido, vemos que o número de ocorrências é menor que no ambiente *clean,* havendo um número significativo de acessos a portos com valores inferiores a 10000 e existindo também pequenos picos para valores mais

elevados do porto de destino, como os existentes no caso aqui apresentado para os intervalos próximos dos valores 33000 e 49000.

Para **o modo *normal* no ambiente *port scan* 1 para 1** a distribuição que melhor se adequa é do tipo *Fatique Life*.

Apesar desta distribuição ser diferente daquela que melhor se adequa ao modo *aggressive* deste ambiente, a semelhança na forma como o número de acessos a cada porto se distribui é enorme, como se pode ser na figura 37. Também para o modo *normal* cerca de 35% dos portos de destino têm valores correspondentes ao primeiro intervalo do histograma, havendo relevância para o número de vezes que os portos abaixo de 10000 são contabilizados e havendo alguns picos, como no caso anterior, para valores mais elevados, nomeadamente portos com números próximos de 33000 e 49000.

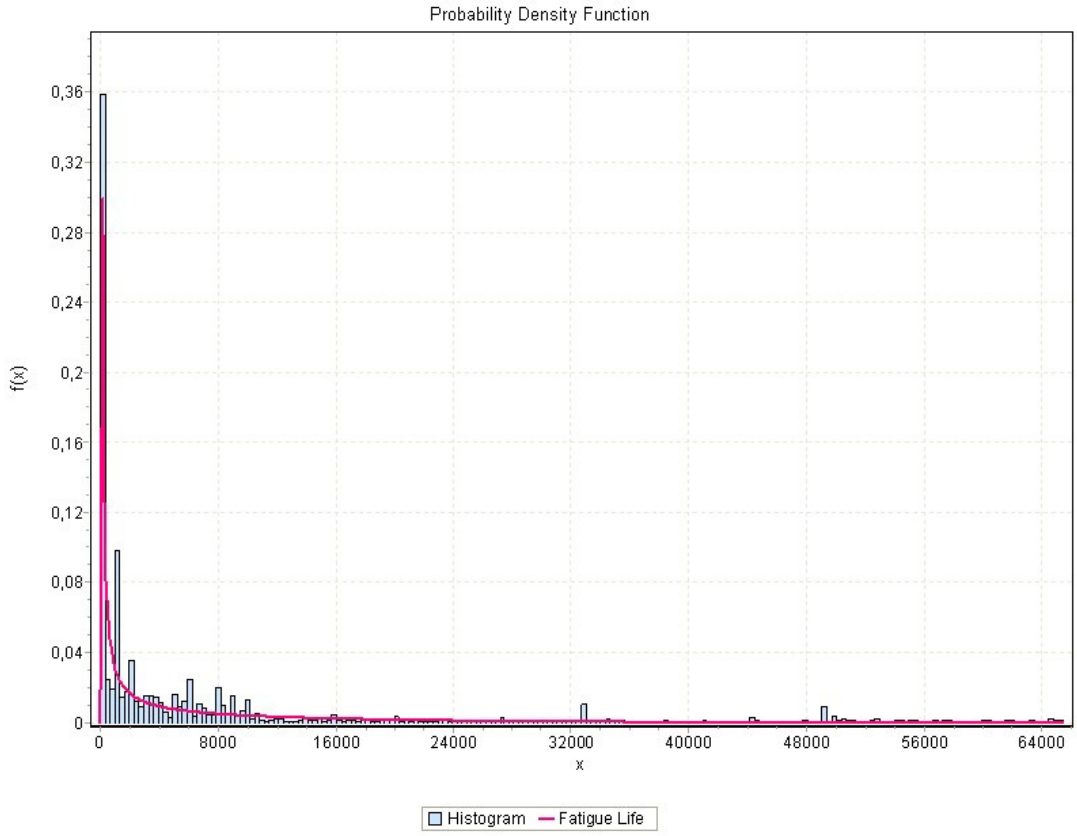

Figura 37 – FDP para os portos de destino no ambiente *port scan* 1 para 1 modo *normal*

A melhor distribuição dos portos de destino quando só as *flags* SYN estão activas, para o **modo *sneaky* do ambiente de *port scan* 1 para 1**, é do tipo *Phased Bi-Exponential*, ou seja, o mesmo tipo apresentado pelo ambiente *clean*, tal como se pode ver na figura 38. Tal como no ambiente *clean,* também cerca de 70% dos portos de destino têm valores compreendidos entre 1 e 320 (pela mesma razão apresentada na sub-secção 5.1.3), sendo que os restantes valores de portos têm percentagens de utilização bastante baixas, quase próximas de zero, e uniformemente distribuídas.

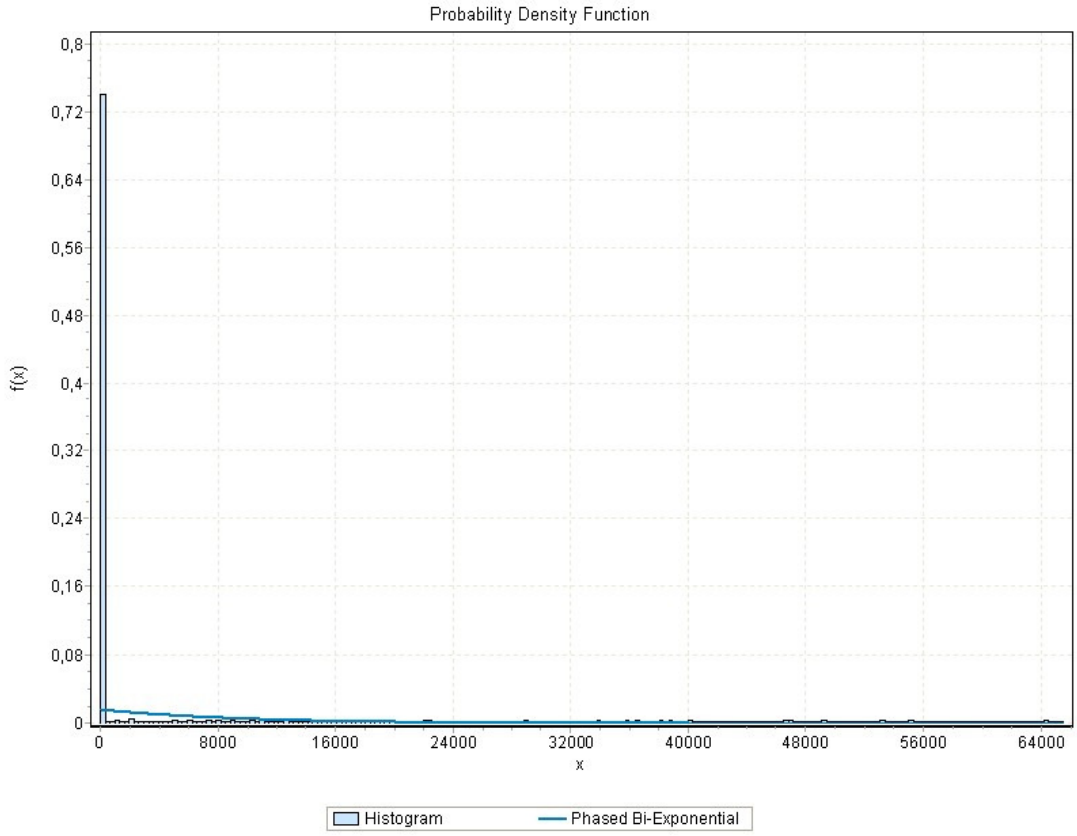

Figura 38 – FDP para os portos de destino no ambiente *port scan* 1 para 1 modo *sneaky*

5.3 – Ambiente *Port scan* 1 para N

Como anteriormente referido, neste ambiente não existem dados para análise no modo *aggressive*, havendo apenas dados para os modos *normal* e *sneak*.

5.3.1 - Número de *flags* SYN por segundo

O **modo *normal*** deste ambiente, para o caso do número de ***flags* SYN isoladas**, apresenta como distribuição que melhor se adapta à sua função de densidade de probabilidade uma distribuição do tipo *Burr*, descrita no anexo C7.

Na figura 39 é possível observar um dos gráficos obtidos para esta distribuição.

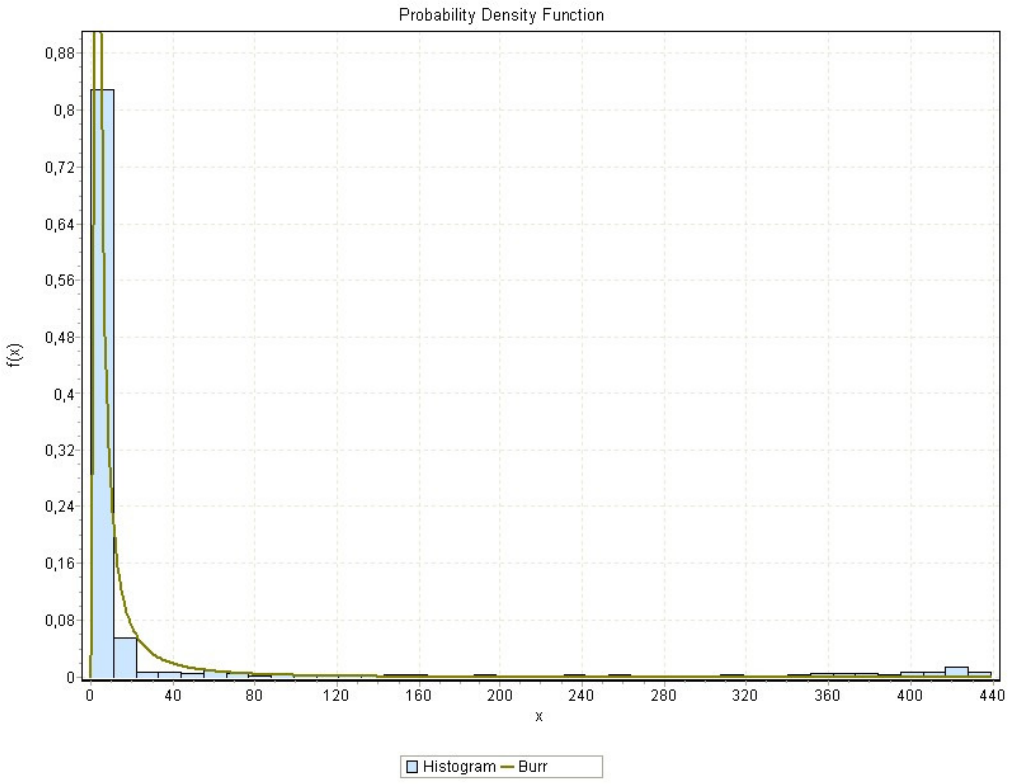

Figura 39 – FDP para *flags* SYN isoladas no ambiente *port scan* 1 para N modo *normal*

Em todos os casos analisados, a grande maioria das contagens de *flags* SYN isoladas por segundo apresentam valores baixos, contidos no primeiro intervalo do histograma (entre 0 e 10); contudo, existem algumas contagens com valores elevados, chegando a ultrapassar as 400 *flags* SYN isoladas por segundo. Este comportamento é distinto daquele que era apresentado pelo ambiente *clean*.

A contagem do número de *flags* SYN isoladas por segundo para o **modo *sneaky*** neste ambiente apresenta uma FDP com distribuição igual à apresentada para o ambiente *clean*, ou seja, distribuição do tipo Johson SB.

Na figura 40 está ilustrado um dos gráficos obtidos. É possível observar através do gráfico apresentado, representativo de todos os casos analisados, que o tipo de distribuição deste modo é igual ao ambiente *clean* e a forma como se distribuem as contagens é também semelhante, sendo apenas perceptível um maior número de contagens para valores baixos, nos intervalos de 0 a 1 e de 1 a 2. O número máximo de contagens por segundo não excede as 15 *flags* por segundo.

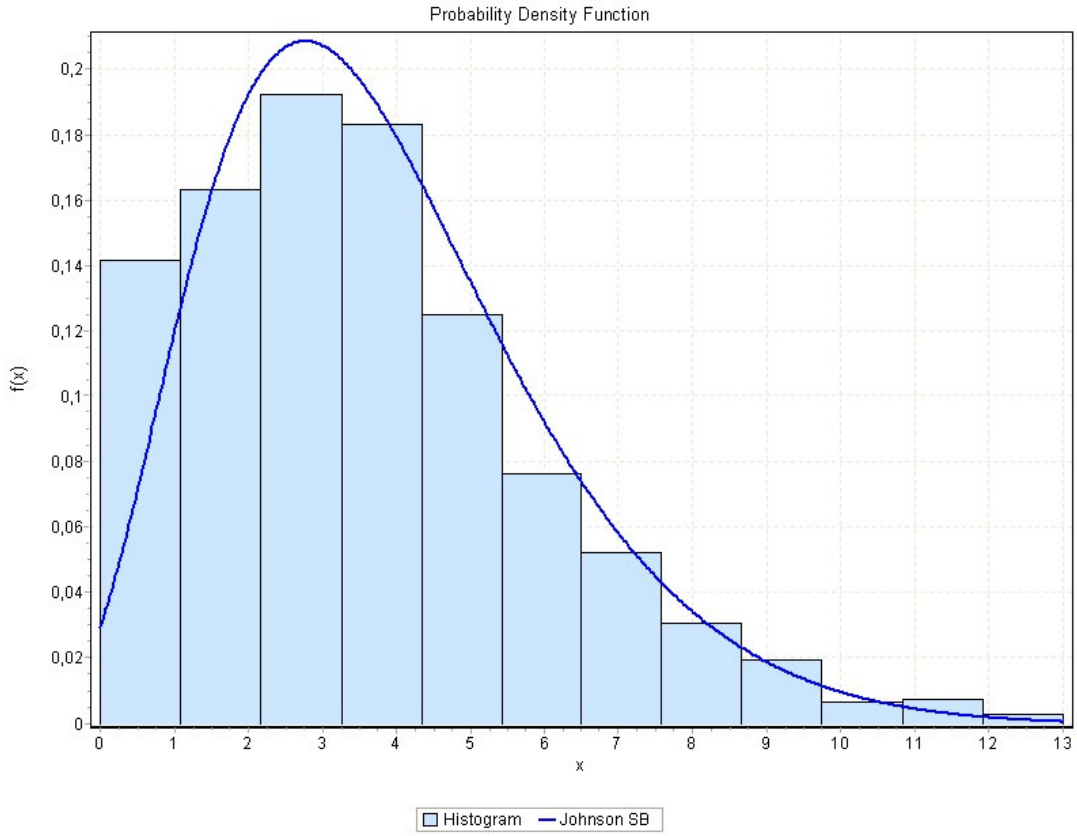

Figura 40 – FDP para *flags* SYN isoladas no ambiente *port scan* 1 para N modo *sneaky*

Relativamente ao número de *flags* SYN não isoladas, tanto o modo *normal* como *sneaky* apresentam a mesma distribuição no caso das *flags* SYN isoladas.

Na figura 41 está ilustrado um dos casos analisados para o número de *flags* SYN não isoladas por segundo no modo *normal* , podendo-se observar que continuam a existir contagens na casa das 400 *flags* por segundo, mas que a grande maioria das contagens por segundo se situa no intervalo de 1 a 10, havendo um aumento no intervalo de 11 a 20.

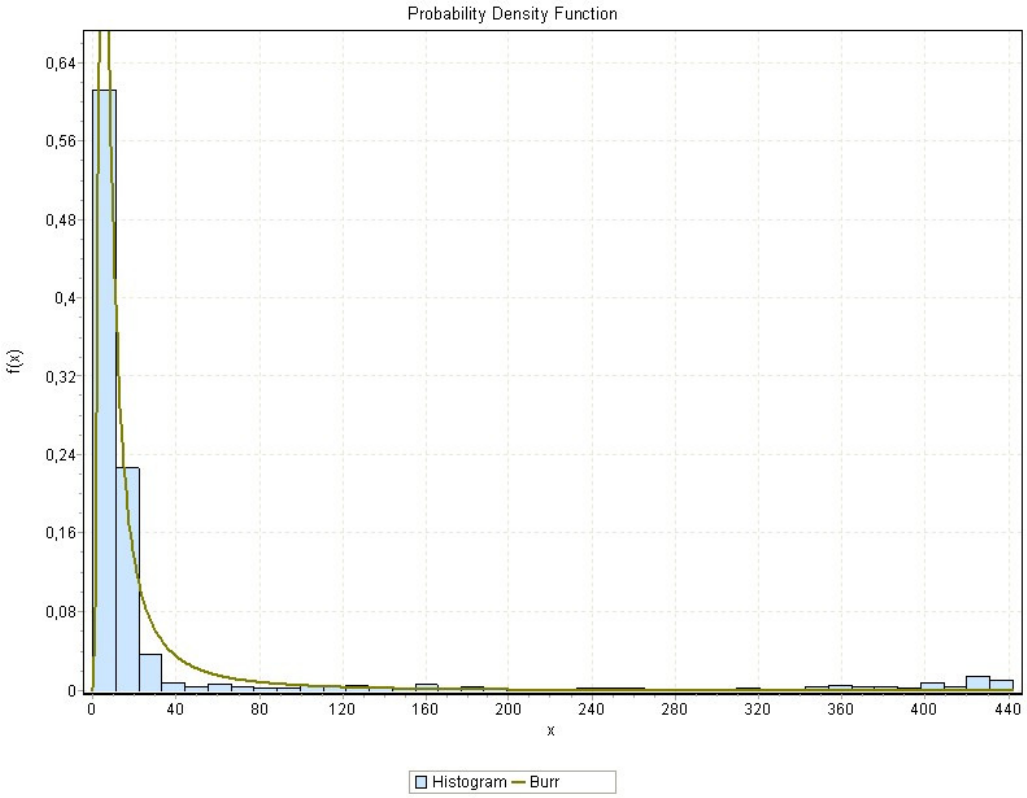

Figura 41 – FDP para *flags* SYN não isoladas no ambiente *port scan* 1 para N modo *normal*

A figura 42 diz respeito ao gráfico de densidade de probabilidade para o número de *flags* SYN não isoladas por segundo no ambiente *port scan* 1 para N no modo *sneaky*. Comparando com o a figura 40, a diferença que mais se destaca é o facto de haver uma duplicação dos valores no eixo dos x, ou seja, a contagem de *flags* SYN por segundo duplicou. O número máximo de *flags* SYN por segundo é de cerca de 25, mas a maioria das contagens encontra-se no intervalo de 2 a 9 *flags* não isoladas por segundo. Este comportamento é semelhante ao apresentado no ambiente *clean* e, recorrendo ao MatLab, analisou-se a razão entre o número de *flags* SYN não isoladas e o número de *flags* SYN isoladas para cada um dos doze ficheiros de dados. Na Tabela 6 são apresentados esses valores.

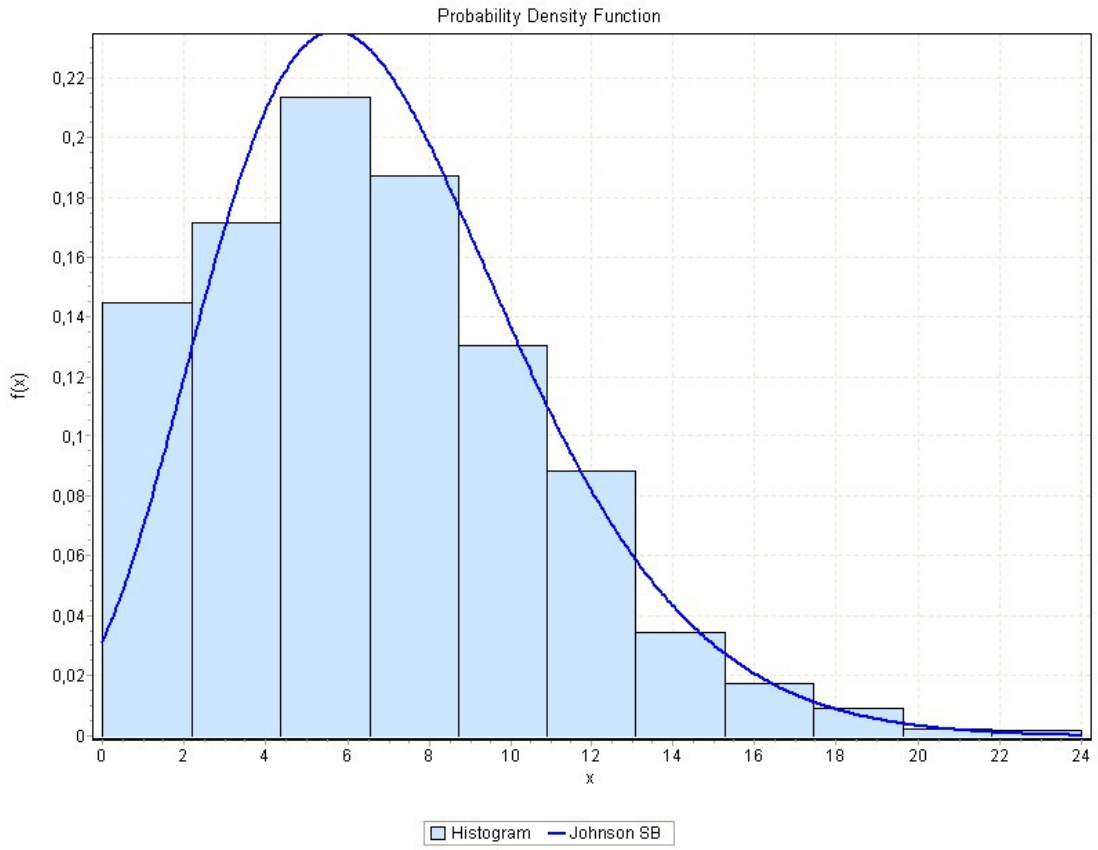

Figura 42 – FDP para *flags* SYN não isoladas no ambiente *port scan* 1 para N modo *sneaky*

Como se pode ver na tabela 6, dos doze casos analisados apenas um apresenta uma razão entre as *flags* SYN não isoladas e isoladas superior a 2, havendo mesmo 9 casos em que essa razão é inferior a 1.90, ou seja, em 11 casos temos uma razão inferior à apresentada pelo ambiente *clean*. Relativamente às médias apresentadas, verifica-se que elas têm valores superiores ao ambiente *clean*.

Número total de *flags* SYN não isoladas.	**Número total de *flags* SYN isoladas.**	**Razão entre número de *flags* SYN não isoladas e isoladas**
10834	5686	1,90538163911361
12762	6919	1,84448619742737
12333	6701	1,84047157140725
12697	6878	1,84603082291364
12587	6831	1,84262919045528
12631	6856	1,84232788798133
12614	6845	1,84280496712929
12445	6758	1,84152116010654
12541	6811	1,84128615474967
12392	6573	1,88528830062376
12346	6131	2,01370086445931
12252	6325	1,93707509881423
Média		
12369,5	6609,5	1,871472879945532

Tabela 6 – Razão entre número de *flags* SYN não isoladas e isoladas para o ambiente *port scan* 1 para N modo *sneaky*

5.3.2 – Intervalo de tempo entre pacotes

A limitação descrita no ponto 5.1.2, imposta pelo EasyFit, também é aqui aplicada.

Os dois modos existentes, ***normal*** e ***sneaky***, para o **ambiente *port scan* 1 para N** apresentam, à semelhança do ambiente *clean* e dos restantes ambientes de *port scan* a distribuição *Fatique Life* como distribuição que melhor ajusta a função densidade de probabilidade.

A quase totalidade dos pacotes chega com um intervalo de tempo entre si inferior a 0.005 segundos, como se pode ver nas figuras 43 e 44 que representam, respectivamente, os gráficos das funções de densidade de probabilidade para os modos *normal* e *sneaky*.

Na tabela 5 estão representadas as médias dos intervalos dos dois casos apresentados nas figuras

	Média do intervalo de tempo entre pacotes
Sneaky	0,00123
Normal	0,00123

Tabela 7 – Média do intervalo de tempo entre pacotes no ambiente *port scan* 1 para N

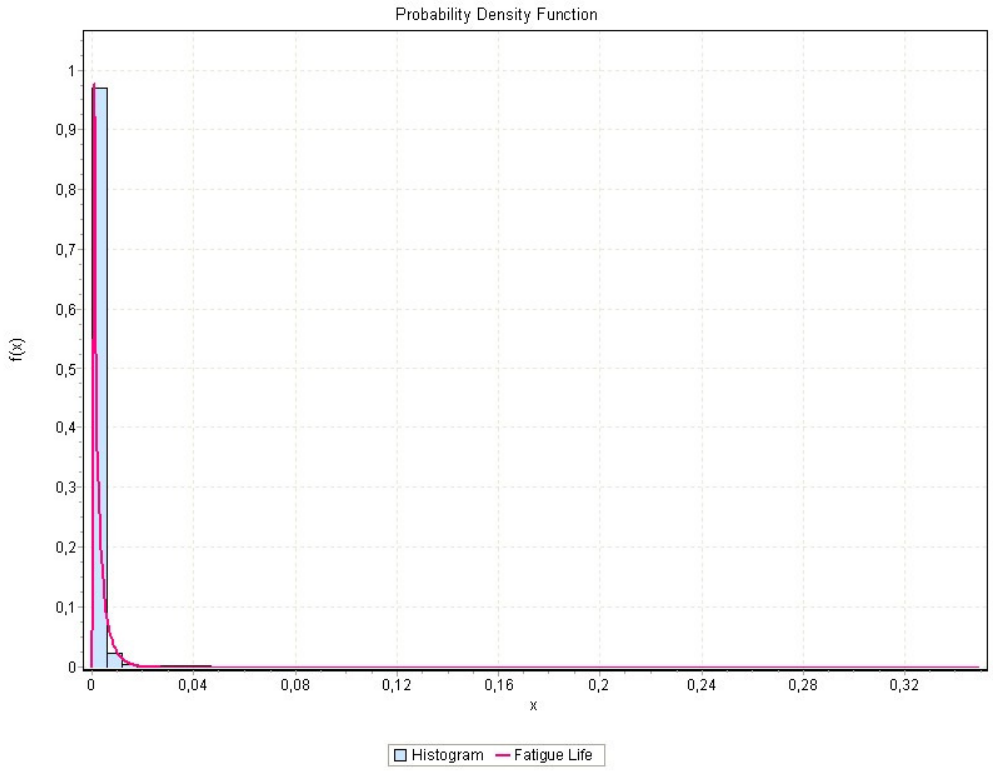

Figura 43 – FDP para o intervalo de tempo entre pacotes no ambiente *port scan* 1 para N modo *normal*

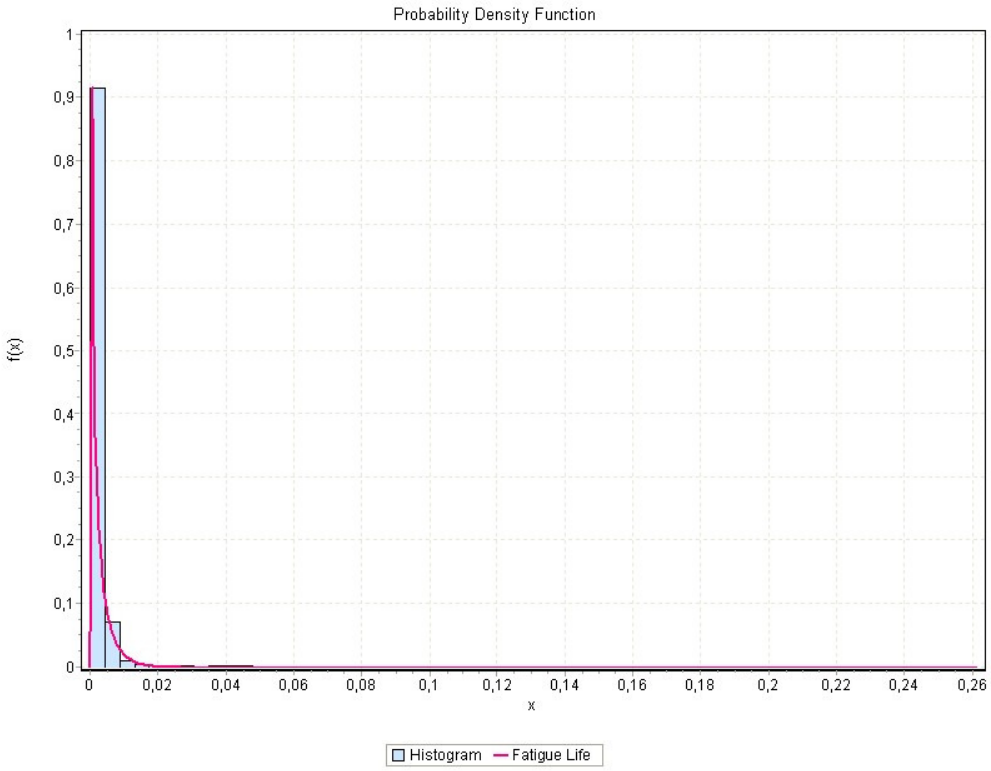

Figura 44 – FDP para o intervalo de tempo entre pacotes no ambiente *port scan* 1 para N modo *sneaky*

5.3.3 – Portos TCP quando só as *flags* SYN se encontram activas

Os gráficos das funções de densidade de probabilidade apresentados são construídos com base em histogramas que dividem o intervalo dos dados no número máximo de intervalos permitido pelo EasyFit, ou seja, 200 intervalos.

Os dois modos existentes, ***normal* e *sneaky***, para o ambiente de ***port scan* 1 para N** apresentam uma distribuição do tipo *Wakeby* como aquela que melhor ajusta a função densidade de probabilidade para os **portos de origem** quando só as *flags* SYN estão activas. Nas figuras 45 e 46 estão presentes os gráficos para os casos *normal* e *sneaky*, respectivamente.

Em ambos os modos, os portos de origem são sempre superiores a 32000 e, apesar de possuírem a mesma distribuição, é possível observar as diferenças entre ambos. No modo *normal* existem vários picos, bastante superiores aos restantes, no número de acessos a alguns intervalos dos portos de origem, sendo que os restantes intervalos apresentam percentagens de acesso muito baixas (cerca de 0.002%). No modo *sneaky* existe apenas um pico, cerca de 20 vezes superior aos restantes intervalos, no último intervalo do gráfico (entre aproximadamente 65200 e 65536). Os restantes intervalos apresentam valores mais ou menos constantes entre si, e na casa dos 0,05% de acessos para cada intervalo. Apesar de as FDP apresentadas nestes ambiente serem iguais às apresentadas no ambiente *clean,* as diferenças entre ambos os ambientes são claramente visíveis.

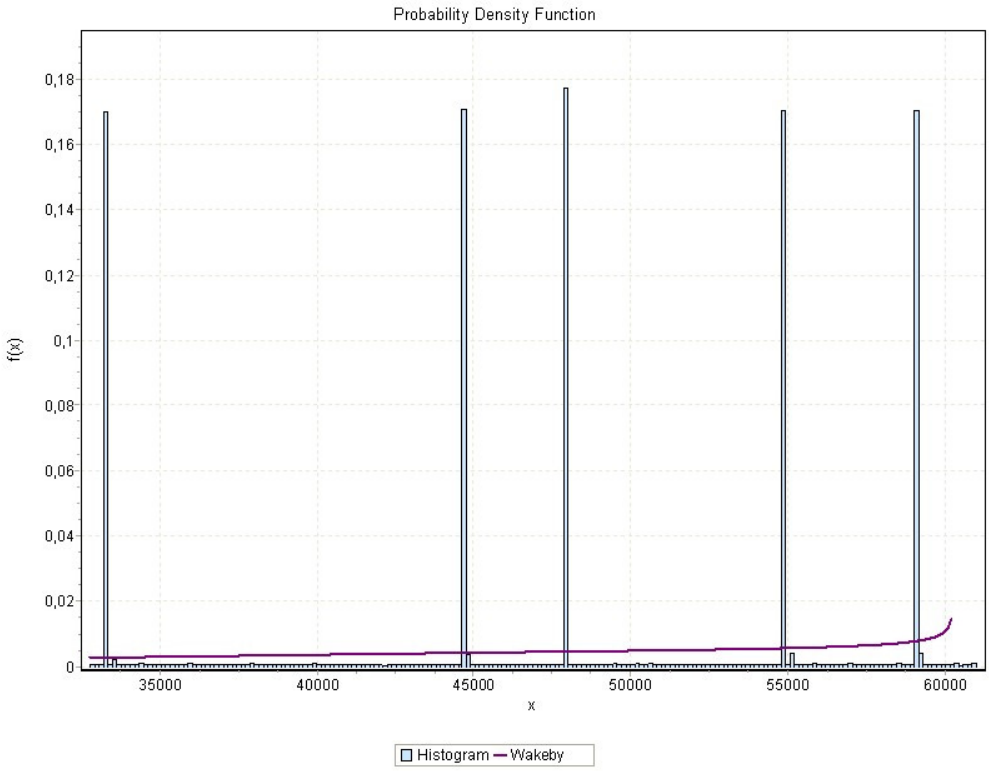

Figura 45 – FDP para os portos de origem no ambiente *port scan* 1 para N modo *normal*

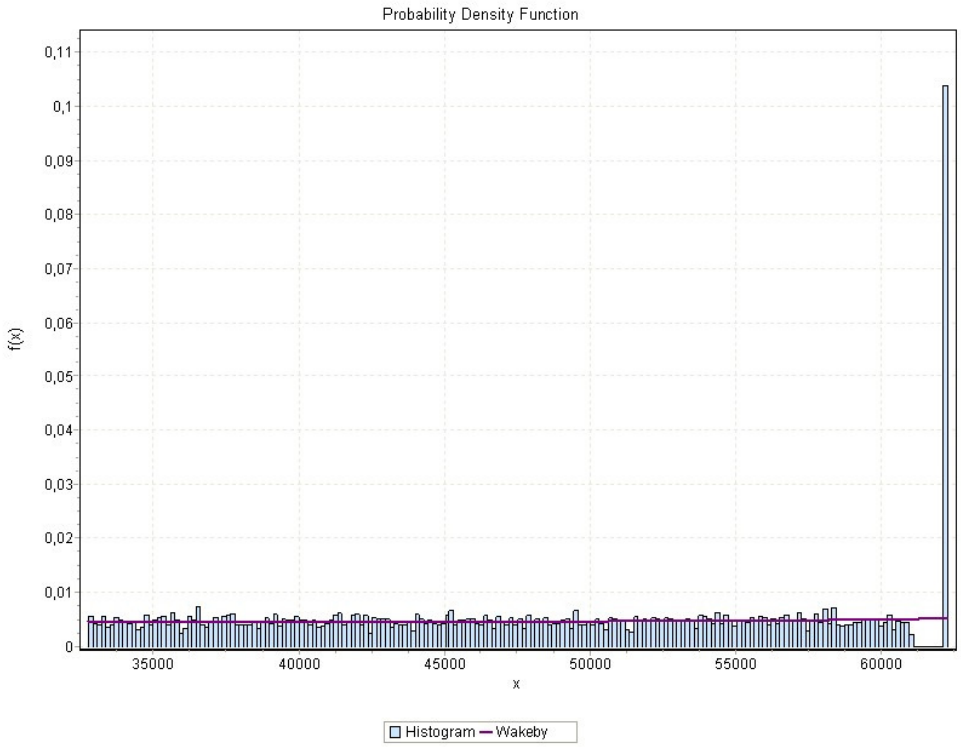

Figura 46 – FDP para os portos de origem no ambiente *port scan* 1 para N modo *sneaky*

Em relação aos **portos de destino** quando só as *flags* SYN estão activas no ambiente ***port scan* 1 para N no modo *normal***, uma distribuição do tipo *Dagum* (anexo C8) é a que melhor ajusta a FDP.

Na figura 47 está ilustrado um exemplo, representativo de todos os casos analisados para este ambiente e modo, da função densidade de probabilidade para os portos de destino quando só as *flags* SYN estão activas. Todo o espectro de valores dos portos TCP está presente, sendo que a maior percentagem (cerca de 17%) de acessos ocorre no primeiro intervalo do histograma e mais de 50% dos acessos dá-se nos primeiros 10000 portos, havendo depois pequenos picos para valores superiores do porto de destino. Em comparação com o ambiente *clean,* observa-se uma menor percentagem de acessos no primeiro intervalo do histograma e uma maior percentagem de acessos para portos abaixo do porto 10000.

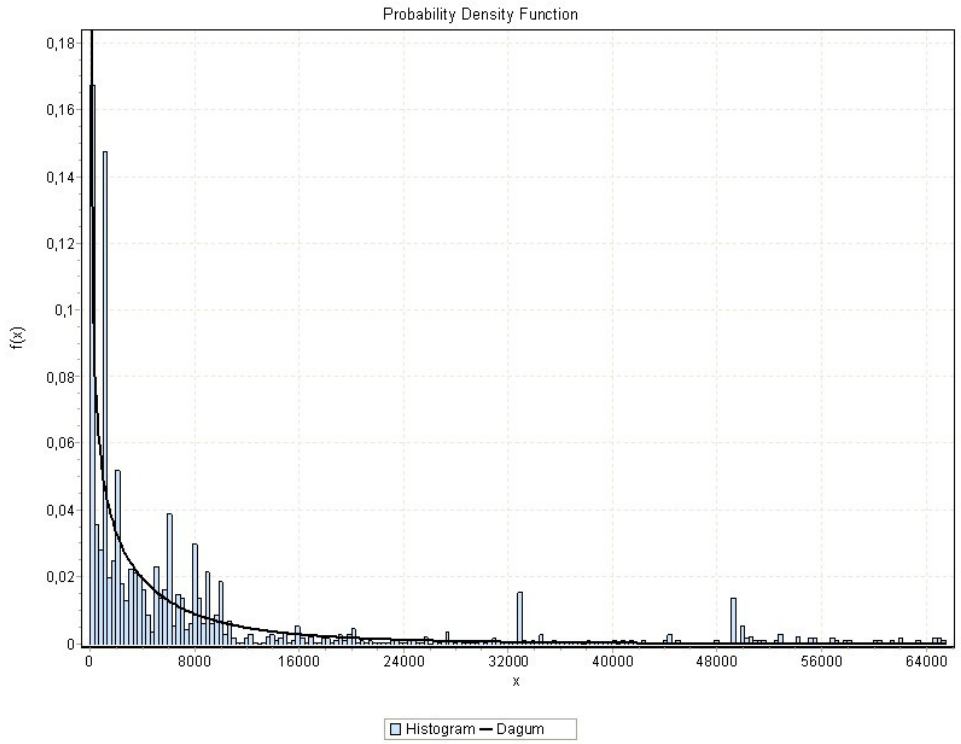

Figura 47 – FDP para os portos de destino no ambiente *port scan* 1 para N modo *normal*

Os portos de destino quando só as *flags* SYN estão activas no **modo *sneaky* do ambiente de *port scan* 1 para N** têm a distribuição *Phased Bi-Weibull*, descrita no anexo C6, como distribuição que melhor ajusta a função densidade de probabilidade. Esta função está ilustrada na figura 48.

A distribuição aqui apresentada é diferente daquela que corresponde ao ambiente *clean*; contudo, as diferenças entre as FDP dos dois ambientes não são imediatas. Como se observa na figura 48, a grande maioria dos portos de destino têm valores entre 1 e aproximadamente 330, sendo que neste ambiente e modo esta percentagem é menor que no ambiente *clean*. Observa-se também algumas pequenas variações nos primeiros intervalos do histograma face aos restantes, que apresentam valores muito baixos. Uma vez mais se nota a influência dos portos mais comuns (1-1023) no valor dos portos de destino e a existência de pequenos picos em torno dos portos com valores próximos de 33000 e 49000.

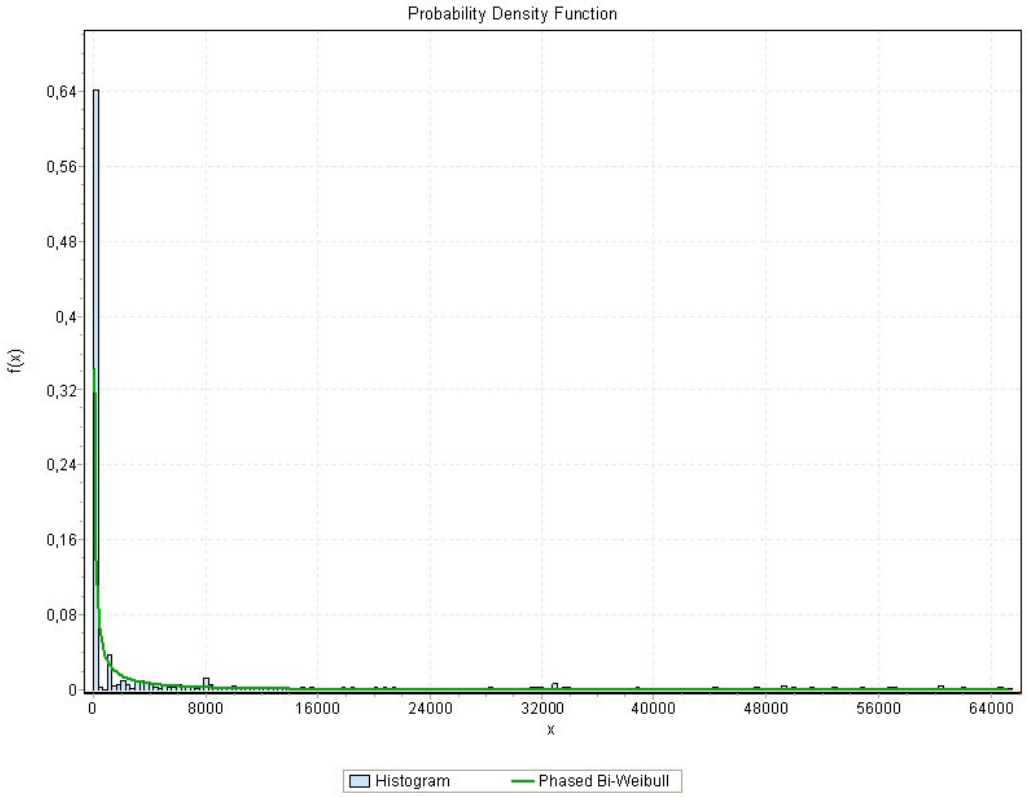

Figura 48 – FDP para os portos de destino no ambiente *port scan* 1 para N modo *sneaky*

5.4 – Ambiente *Port scan* N para 1

Neste ambiente voltamos a ter disponíveis dados referentes aos três modos: *aggressive*, *normal* e *sneaky*.

5.4.1 - Número de *flags* SYN por segundo

O **modo *aggressive*** e o **modo *sneaky*** deste ambiente apresentam como distribuição que melhor ajusta a função densidade de probabilidade para o número de *flags* SYN isoladas por segundo, uma distribuição do tipo *Burr*.

Em ambos os modos é possível observar que a quase totalidade das contagens de *flags* SYN isoladas por segundo apresenta valores bastantes baixos, entre 0 e 10 *flags* isoladas por segundo. Por outro lado, todos os casos analisados, tanto no modo *aggressive* como no modo *sneaky*, apresentam contagens que ultrapassam as 400 *flags* isoladas por segundo, sendo este comportamento distinto daquele que era apresentado pelo ambiente *clean*.

Na figura 49 está ilustrado um dos gráficos da FDP da contagem de *flags* SYN isoladas por segundo, obtido para o modo *aggressive,* enquanto que na figura 50 está representado o gráfico correspondente ao modo *sneaky*.

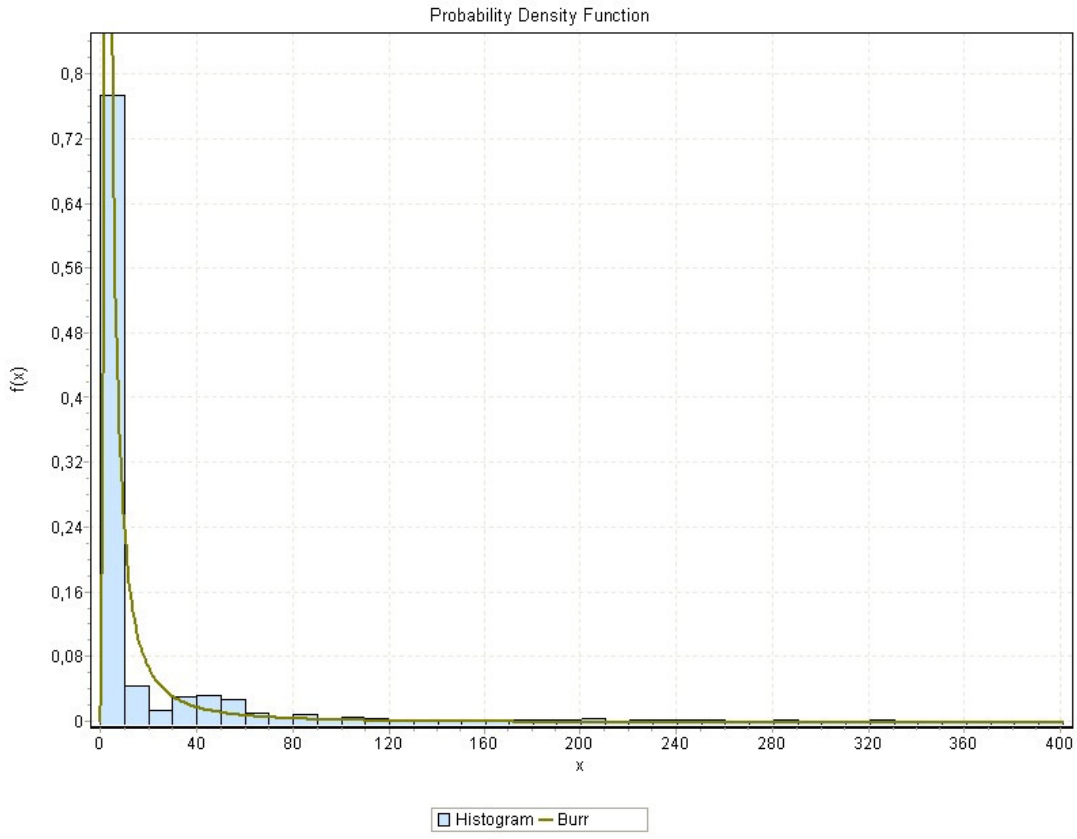

Figura 49 – FDP para *flags* SYN isoladas no ambiente *port scan* N para 1 modo *aggressive*

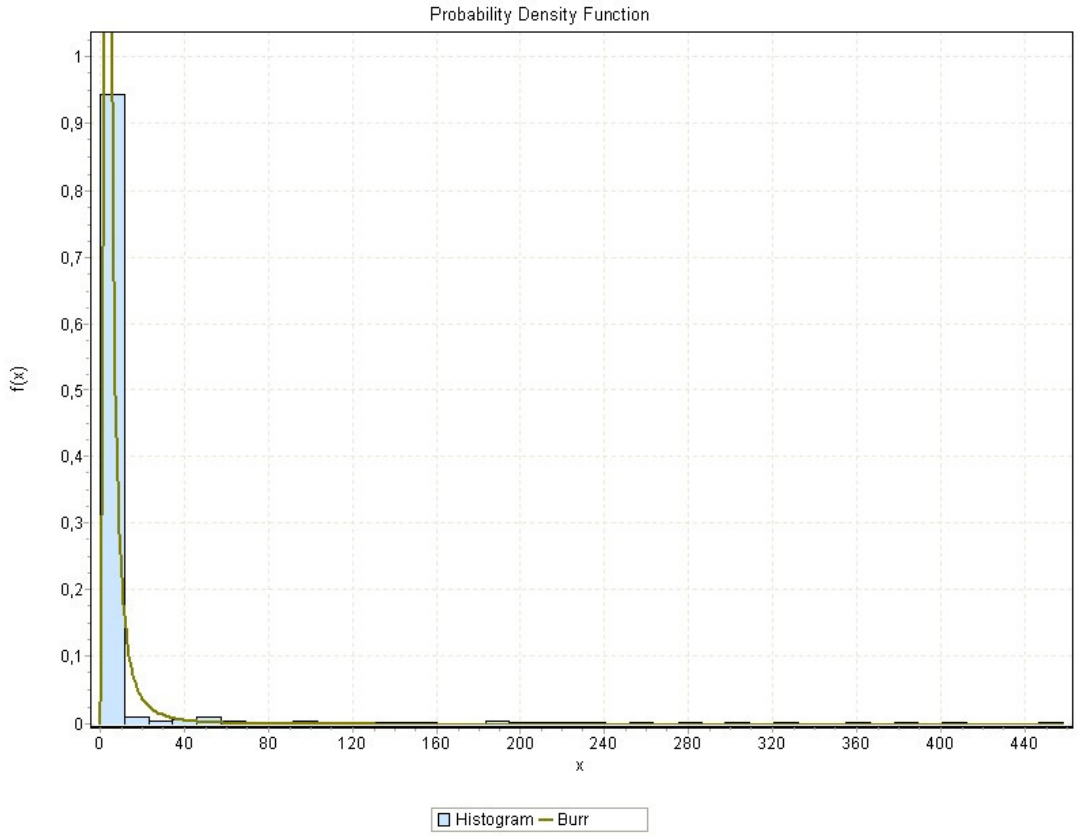

Figura 50 – FDP para *flags* SYN isoladas no ambiente *port scan* N para 1 modo *sneaky*

O **modo *normal*** do ambiente ***port scan* N para 1** apresenta, na análise do número de *flags* SYN isoladas por segundo, algumas diferenças face aos dois outros modos. Começando pela distribuição que melhor ajusta os dados, que para o modo *normal* é uma distribuição do tipo *Wakeby* (em anexo C3).

Na figura 51 está ilustrado um dos gráficos da FDP obtidos para o modo *normal*.

É possível observar, em todos os casos analisados, que cerca de 30% das contagens se encontra no intervalo de 0 a 10 *flags* SYN isoladas por segundo e que a contagem até cerca de 120 *flags* por segundo é relevante, tal como nos modos *aggressive* e *normal,* e diferindo do ambiente *clean*: observam-se contagens de *flags* SYN isoladas acima das 400 por segundo, chegando em alguns casos a valores perto das 1000 *flags* isoladas por segundo.

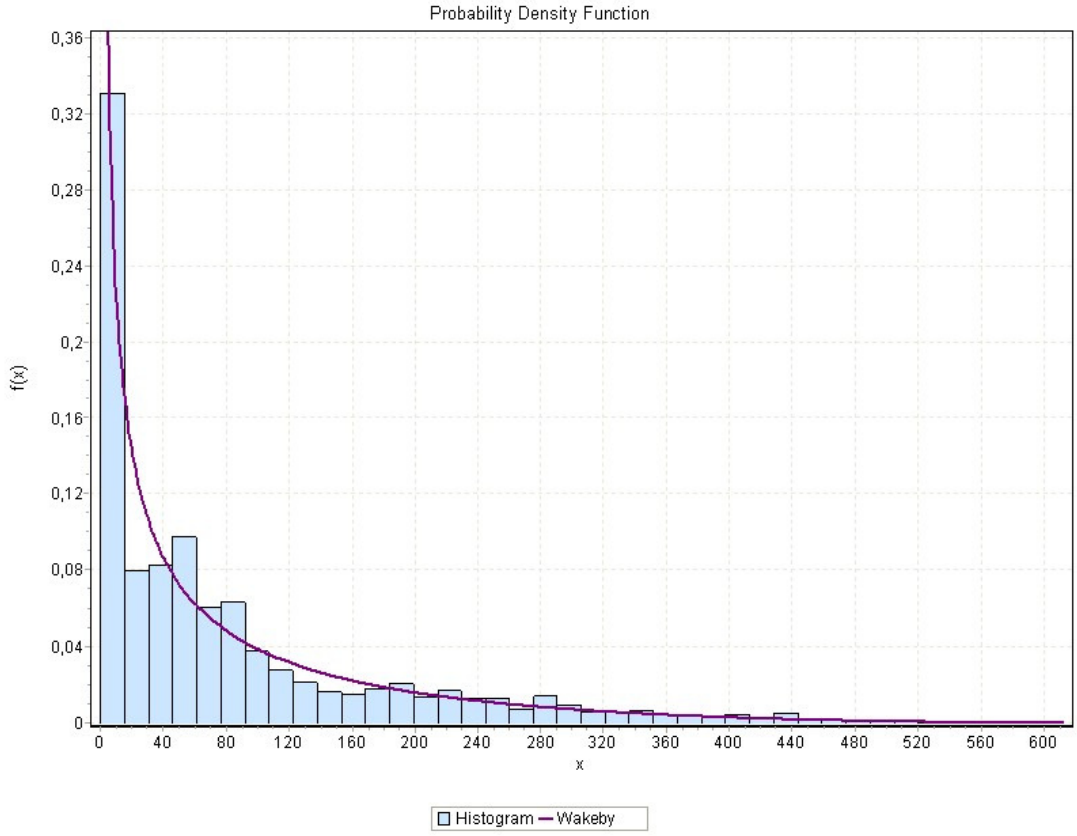

Figura 51 – FDP para *flags* SYN isoladas no ambiente *port scan* N para 1 modo *normal*

A análise no número de *flags* SYN não isoladas por segundo revela, no ambiente ***port scan* N para 1** nos modos ***aggressive*** e ***sneaky***, que apresentam a mesma distribuição e um comportamento semelhante ao do número de *flags* SYN isoladas por segundo, ou seja, uma distribuição do tipo *Wakeby*. Nas figuras 52 e 53 estão ilustrados dois dos gráficos da FDP para os modos *aggressive* e *sneaky,* respectivamente.

Observa-se que a grande maioria das contagens se mantém no intervalo de 0 a 10 *flags* não isoladas por segundo, havendo um aumento da contagem no intervalo seguinte (11 a 20 *flags* não isoladas por segundo). Em ambos os modos se observam contagens de *flags* SYN não isolados acima das 400 por segundo, diferindo assim do ambiente *clean*.

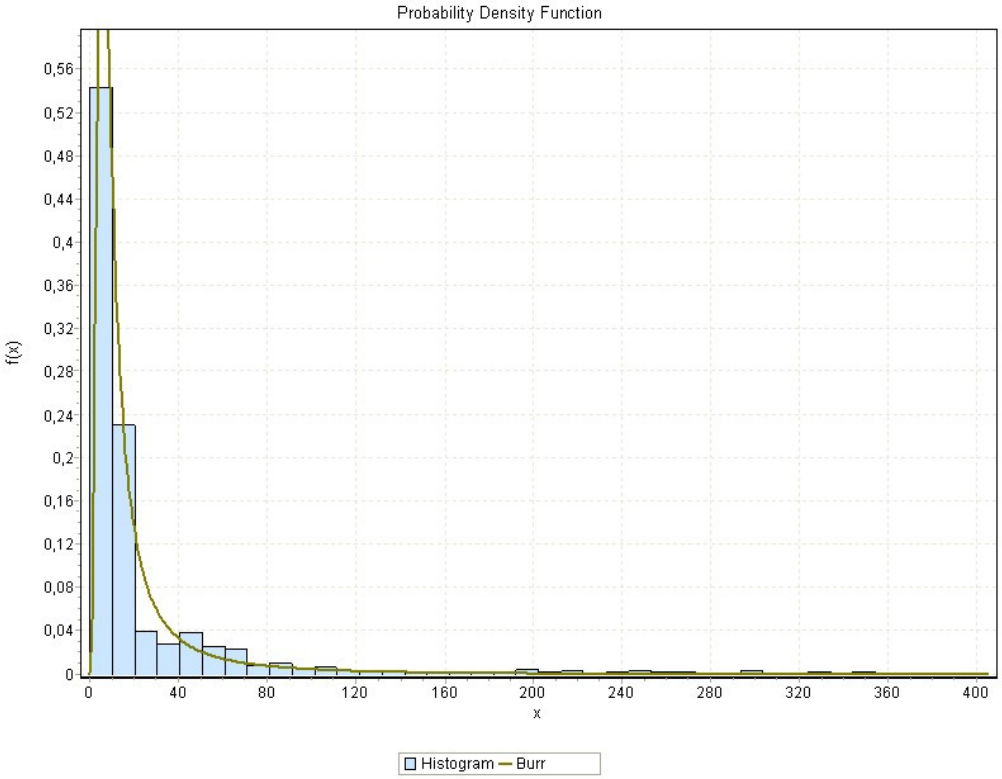

Figura 52 – FDP para *flags* SYN não isoladas no ambiente *port scan* N para 1 modo *aggressive*

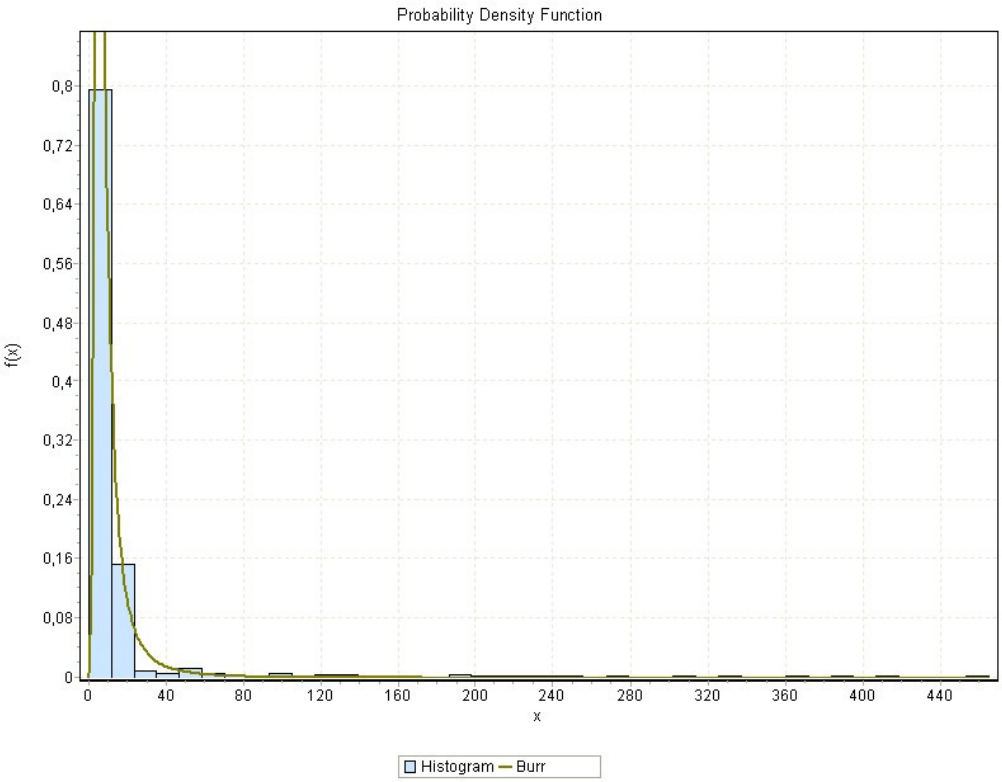

Figura 53 – FDP para *flags* SYN não isoladas no ambiente *port scan* N para 1 modo *sneaky*

O **modo *normal*** deste ambiente, quando se analisa o número de *flags* SYN não isoladas por segundo, apresenta uma distribuição do tipo *Lognormal* (descrita no anexo C9).

Na figura 54 está ilustrado um dos gráficos da FDP com distribuição *Lognormal* obtidos para este modo.

É visível uma maior percentagem de *flags* SYN não isoladas por segundo no primeiro intervalo do histograma, aproximadamente entre 0 e 20, havendo depois uma curva descendente para os restantes intervalos e estabilizando perto das 400 *flags* SYN não isoladas por segundo. Chegam a existir contagens próximas das mil *flags* não isoladas por segundo (no exemplo da figura 54 mais de 640).

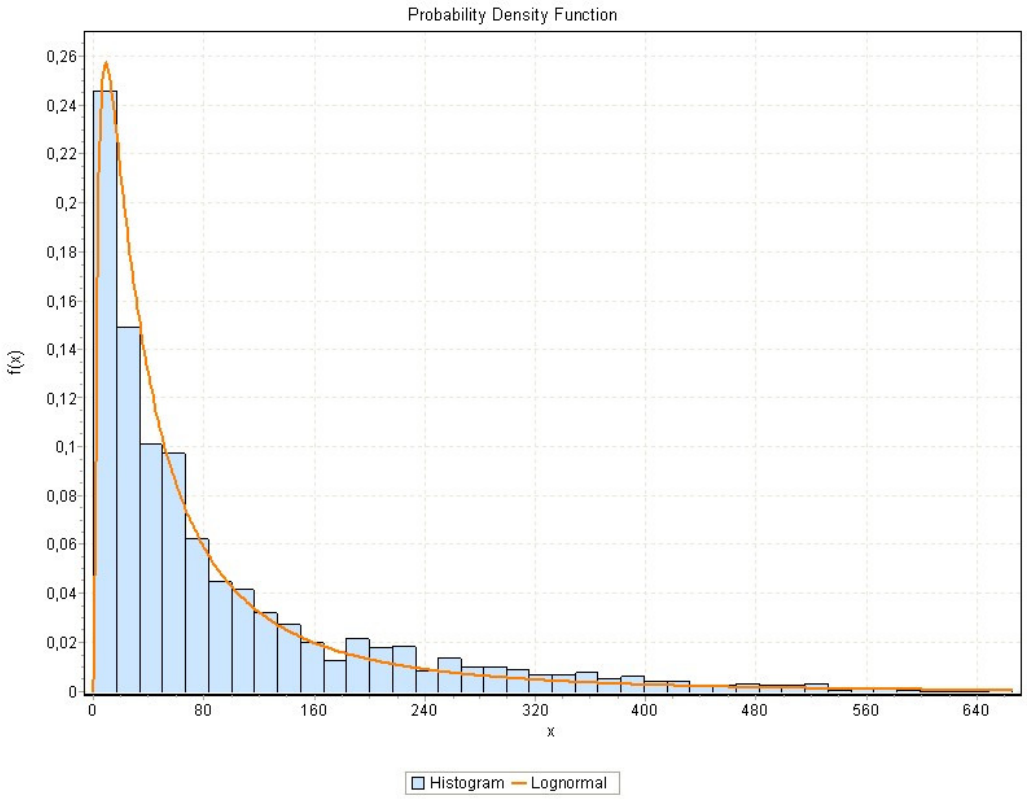

Figura 54 – FDP para *flags* SYN não isoladas no ambiente *port scan* N para 1 modo *normal*

Devido às evidentes diferenças entre o ambiente *port scan* N para 1 e o ambiente *clean,* não se calculou a razão entre as *flags* SYN não isoladas e isoladas para qualquer um dos três modos.

5.4.2 – Intervalo de tempo entre pacotes

A limitação referente à quantidade de dados suportada pelo EasyFit, descrita no ponto 5.1.2, também aqui se aplica de igual forma.

Os modos *aggressive*, *normal* e *sneaky* deste ambiente apresentam todos uma distribuição para o intervalo entre pacotes que é ajustada por uma função de probabilidade do tipo *Fatique Life*.

Na tabela 8 são apresentadas as médias dos intervalos entre pacotes para cada um dos três modos do ambiente *port scan* N para 1 ilustrados nas figuras 55, 56 e 57, que apresentam os gráficos das funções densidade de probabilidade para os modos *aggressive*, *normal* e *sneaky*, respectivamente.

Nos três modos, a quase totalidade dos pacotes chega com um intervalo de tempo entre si que se enquadra no primeiro intervalo do histograma. Como se pode ver pela tabela 8, as médias dos três modos são muito próximas. Nos três modos, existe uma pequena percentagem de valores pertencentes ao segundo ou mesmo terceiro intervalo do histograma, sendo praticamente nulo nos restantes intervalos. O valor máximo do intervalo entre pacotes não varia significativamente entre os três modos. O ambiente *port scan* N para 1 apresenta comportamento igual ao ambiente clean e restantes ambientes.

	Média do intervalo de tempo entre pacotes
Sneaky	0,00119
Normal	0,00107
Aggressive	0,00118

Tabela 8 – Média do intervalo de tempo entre pacotes no ambiente *port scan* N para 1

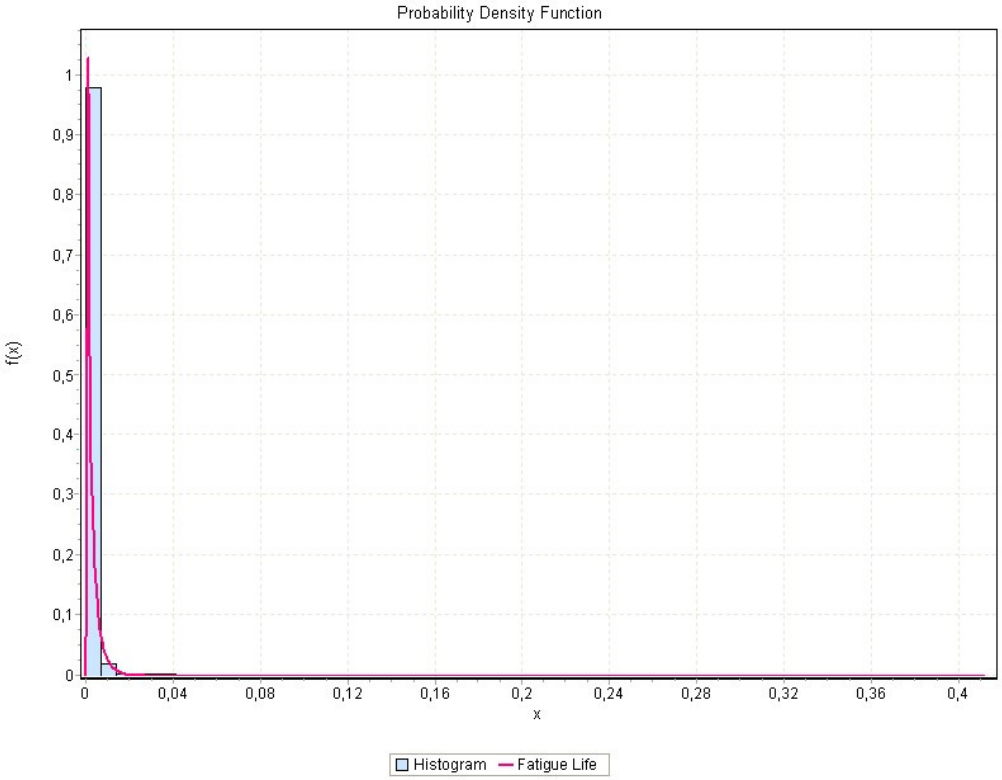

Figura 55 – FDP para o intervalo de tempo entre pacotes no ambiente *port scan* N para 1 modo *aggressive*

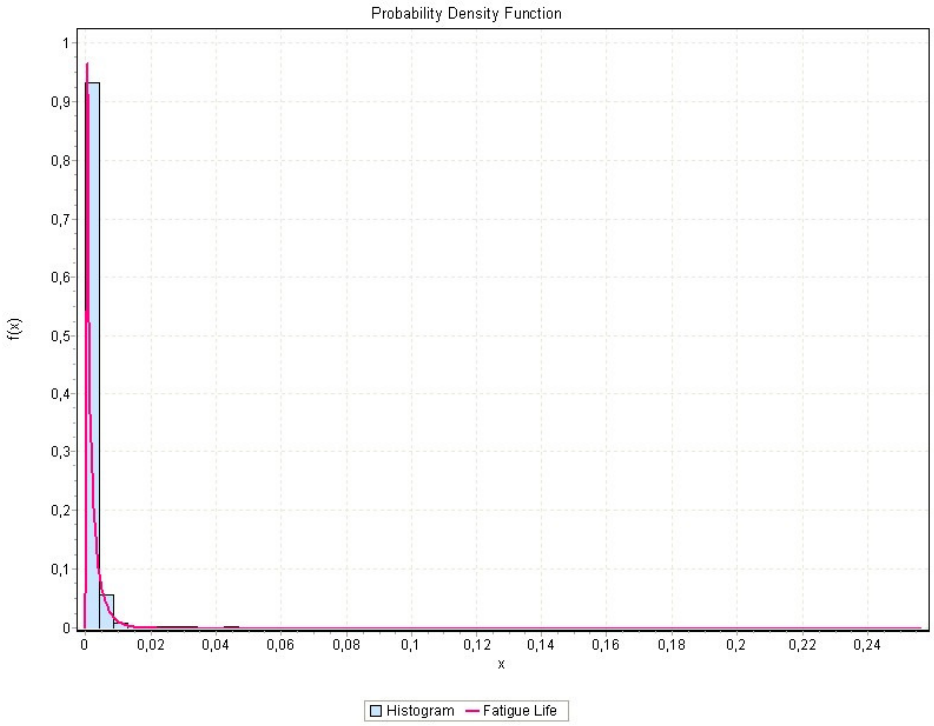

Figura 56 – FDP para o intervalo de tempo entre pacotes no ambiente *port scan* N para 1 modo *normal*

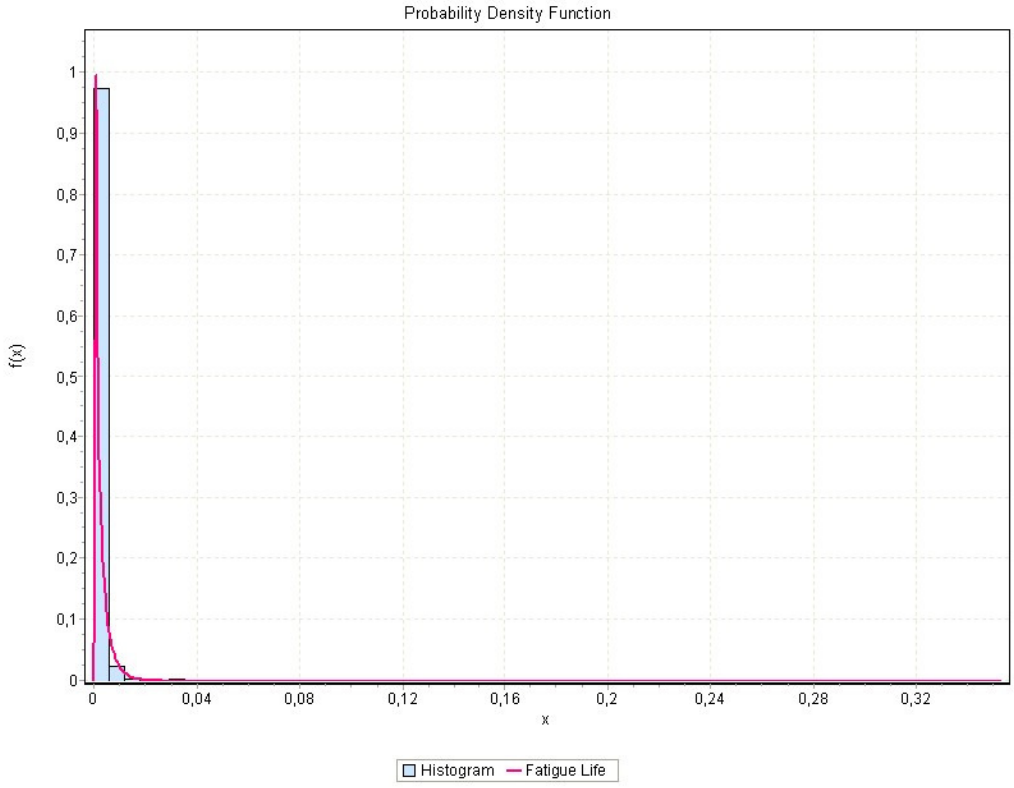

Figura 57 – FDP para o intervalo de tempo entre pacotes no ambiente *port scan* N para 1 modo *sneaky*

5.4.3 – Portos TCP quando só as *flags* SYN se encontram activas

Devido à grande gama de portos existentes, os histogramas para a construção das funções densidade de probabilidade são construídos com o número máximo de intervalos permitidos pelo EasyFit, ou seja, 200.

Os três **modos, *aggressive*, *normal* e *sneaky***, apresentam uma distribuição do tipo *Wakeby* (anexo C3) para a função densidade de probabilidade para os portos de origem quando só as *flags* SYN se encontram activas. Nos três modos, os portos de origem apenas têm valores superiores a aproximadamente 32000 e apresentam diversos picos em vários intervalos do histograma.

O modo *aggressive*, representado na figura 58, apresenta na grande maioria dos intervalos do histograma baixas percentagens, cerca de 0,1%, tendo diversos intervalos que apresentam picos que podem chegar aos 5%, comportamento este distinto do ambiente clean.

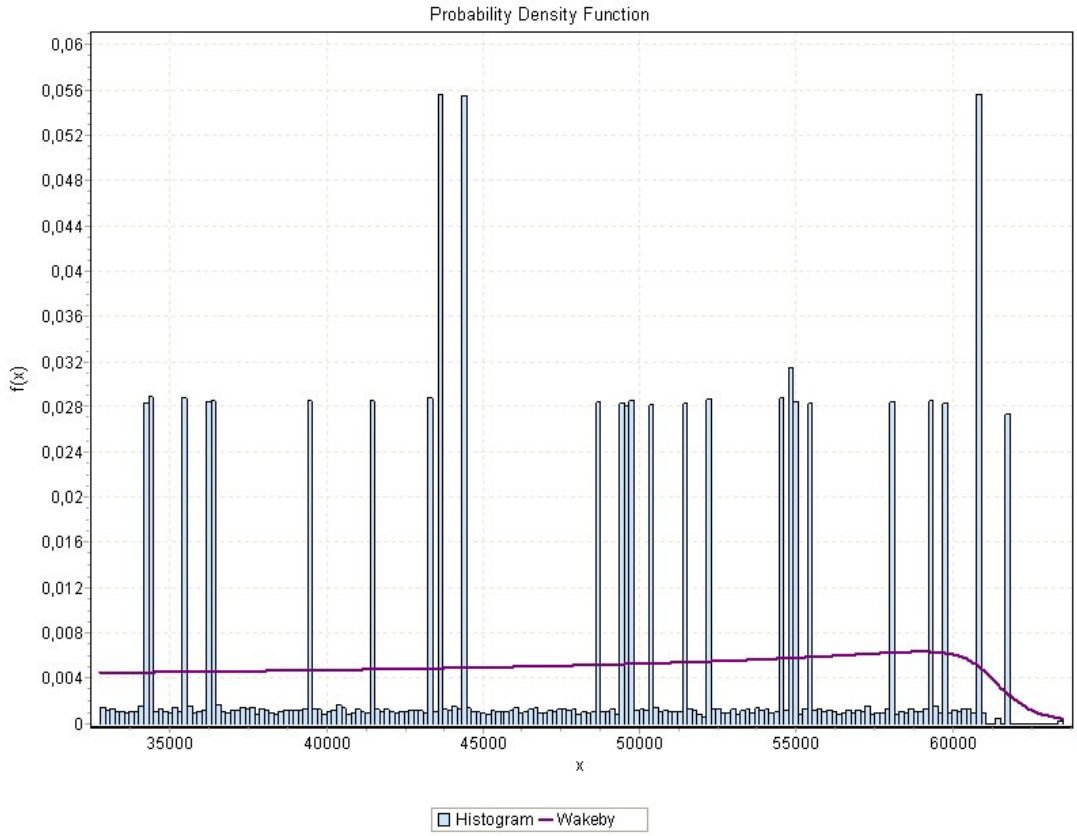

Figura 58 – FDP para os portos de origem no ambiente *port scan* N para 1 modo *aggressive*

O modo *normal,* cujo gráfico da função densidade de probabilidade se encontra na figura 59, apresenta, tal como o modo *aggressive*, diversos picos em vários intervalos do histograma, sendo possível distinguir três níveis: intervalos de portos de origem cuja contagem é próxima de zero (menos de 0,1% por intervalo), intervalos de portos de origem com um número de contagens superior ao nível anterior (cerca de 0,8% por intervalo) e intervalos que apresentam picos com percentagens de contagens superiores aos outros níveis, chegando alguns intervalos a valores próximos dos 3%.

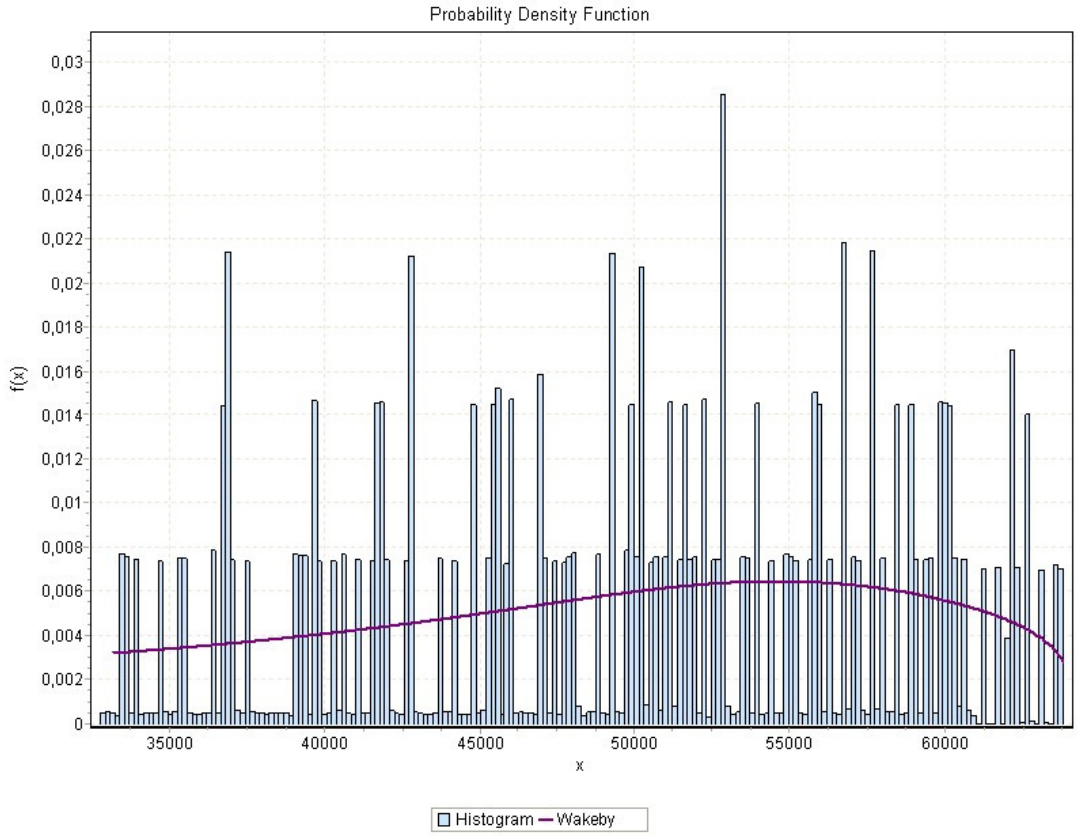

Figura 59 – FDP para os portos de origem no ambiente *port scan* N para 1 modo *normal*

Na figura 60 está ilustrado um exemplo para o gráfico da função densidade de probabilidade dos portos de origem no ambiente *port scan* N para 1 modo *sneaky*. À semelhança dos outros dois modos deste ambiente, o modo *sneaky* apresenta picos em diversos intervalos do histograma, alguns pouco significativos e outros bastante maiores do que a média, chegando acima dos 5% quando a grande maioria dos intervalos tem percentagens abaixo dos 0,5%.

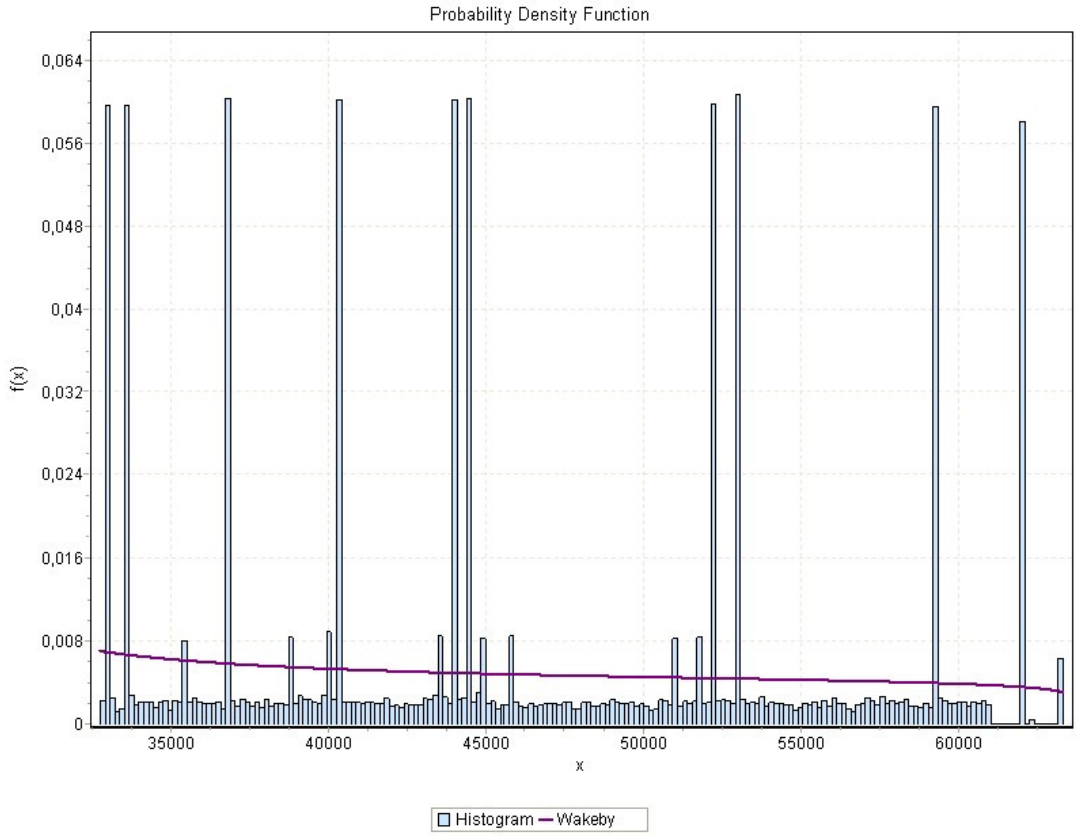

Figura 60 – FDP para os portos de origem no ambiente *port scan* N para 1 modo *sneaky*

Em relação aos dados referentes ao **porto de destino** em ambiente ***port scan* N para 1 modo *aggressive***, a distribuição que melhor se ajusta é a distribuição *Pearson* tipo 6 descrita no anexo C10.

Na figura 61 é possível ver um dos gráficos obtidos para os portos de destino obtidos neste ambiente e modo. Observa-se que cerca de 20% dos pacotes tem como porto de destino um valor que varia entre 1 e 320 aproximadamente, sendo que a quase totalidade dos pacotes tem portos de destino com valores inferiores a 10000; o número de pacotes com portos de destino superiores a 10000 é quase nulo, com excepção de alguns intervalos, nomeadamente em torno dos portos 33000 e 49000.

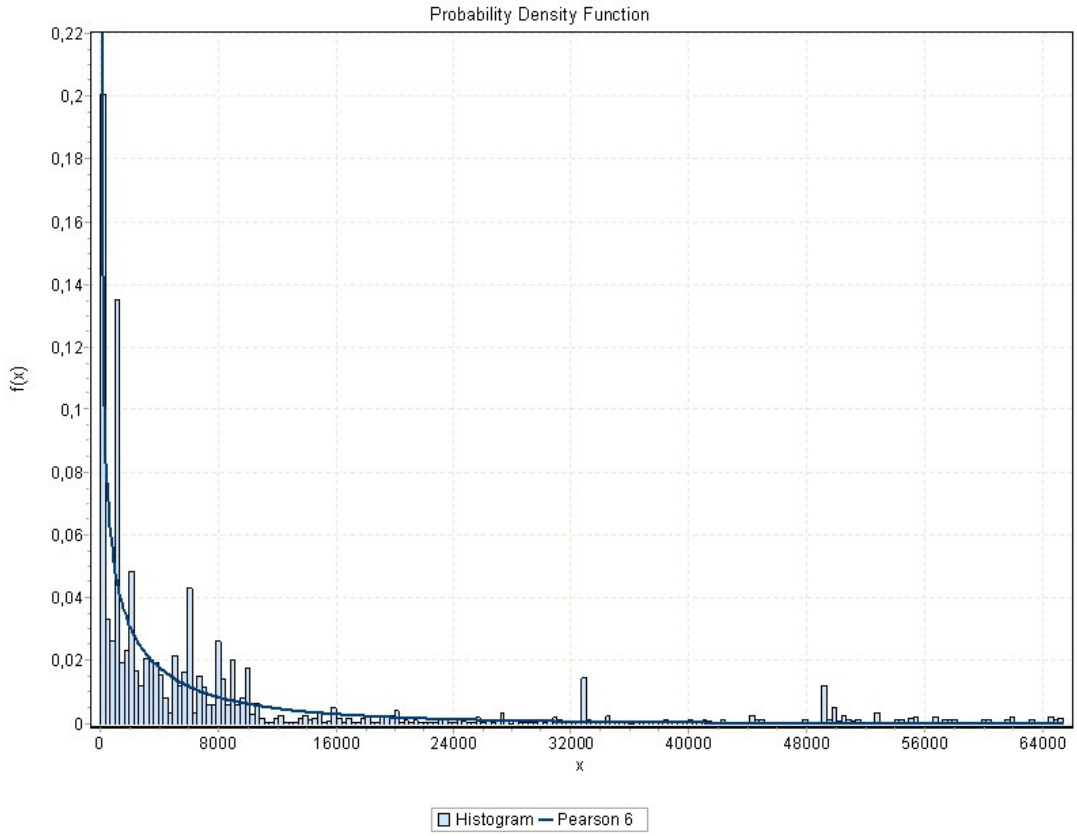

Figura 61 – FDP para os portos de destino no ambiente *port scan* N para 1 modo *aggressive*

O **modo *normal*** deste ambiente apresenta como distribuição que melhor ajusta os dados dos portos de destino uma função de probabilidade *Pareto* tipo 2 (mais detalhes no anexo C11). Na figura 62 está ilustrado um dos gráficos obtidos.

Apesar de modo *normal* apresentar uma distribuição diferente, podemos observar que os gráficos da função densidade de probabilidade são parecidos com os gráficos do modo *aggressive.* Os portos de destino que mais vezes surgem quando só as *flags* SYN estão activas situam-se nos primeiros intervalos do histograma, e correspondem aos portos mais comuns. Existe também alguma relevância na percentagem de portos com número inferior a aproximadamente 10000 e intervalos com picos de cerca de 2% para portos de destino em redor dos números 33000 e 49000.

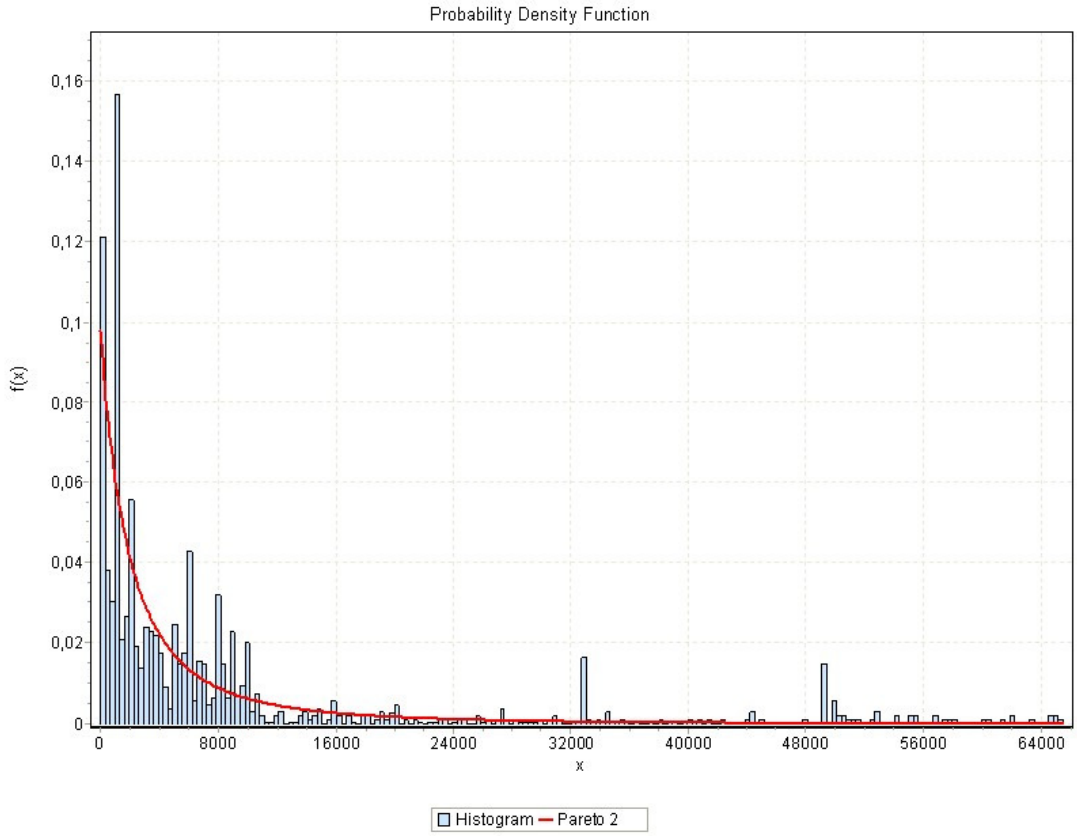

Figura 62 – FDP para os portos de destino no ambiente *port scan* N para 1 modo *normal*

Por fim, os portos de destino quando só as *flags* SYN estão activas no **modo *sneaky*** do ambiente *port scan* N para 1, apresentam uma distribuição do tipo *Gamma*, descrita no anexo C12, como se pode ver através do exemplo da figura 63.

Seguindo a linha dos outros dois modos deste ambiente, o modo *sneaky* apresenta a maior percentagem de contagens para portos no primeiro intervalo do histograma, chegando no exemplo da figura 63 próximo dos 32%; depois, com menos relevância, existe uma curva descendente de portos compreendidos entre o primeiro intervalo do histograma e os portos de destino até aproximadamente 10000. Voltam a destacar-se dois pequenos picos em torno dos valores 33000 e 49000.

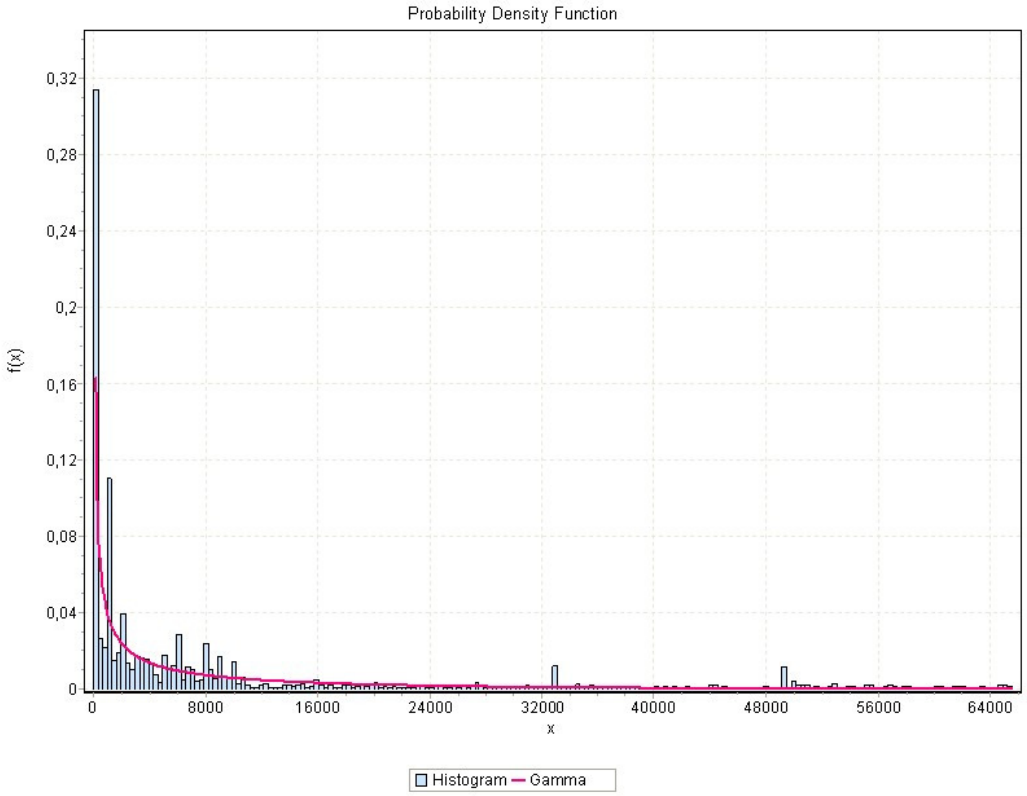

Figura 63 – FDP para os portos de destino no ambiente *port scan* N para 1 modo *sneaky*

5.5 – Ambiente *Port scan* N para N

Como referido anteriormente, no ambiente *port scan* N para N não existem dados para análise no modo *aggressive*, havendo apenas dados para o modo *normal* e *sneaky*.

5.5.1 - Número de *flags* SYN por segundo

O **modo *normal*** do ambiente *port scan* N para N apresenta como distribuição que melhor ajusta o número de *flags* SYN isoladas por segundo uma função de probabilidade do tipo *Fatique Life* (anexo C2).

Na figura 64 está um dos gráficos da FDP obtido e representativo dos restantes. Em todos os casos analisados é evidente o elevado número de contagens de *flags* SYN isoladas por segundo, entre 10 e 20. A partir das 20, o número de *flags* isoladas por segundo vai decrescendo até ser aproximadamente nulo. Apesar da grande maioria das contagens ter um número baixo de *flags* isoladas por segundo, existem contagens que chegam perto de 800 *flags* isoladas por segundo.

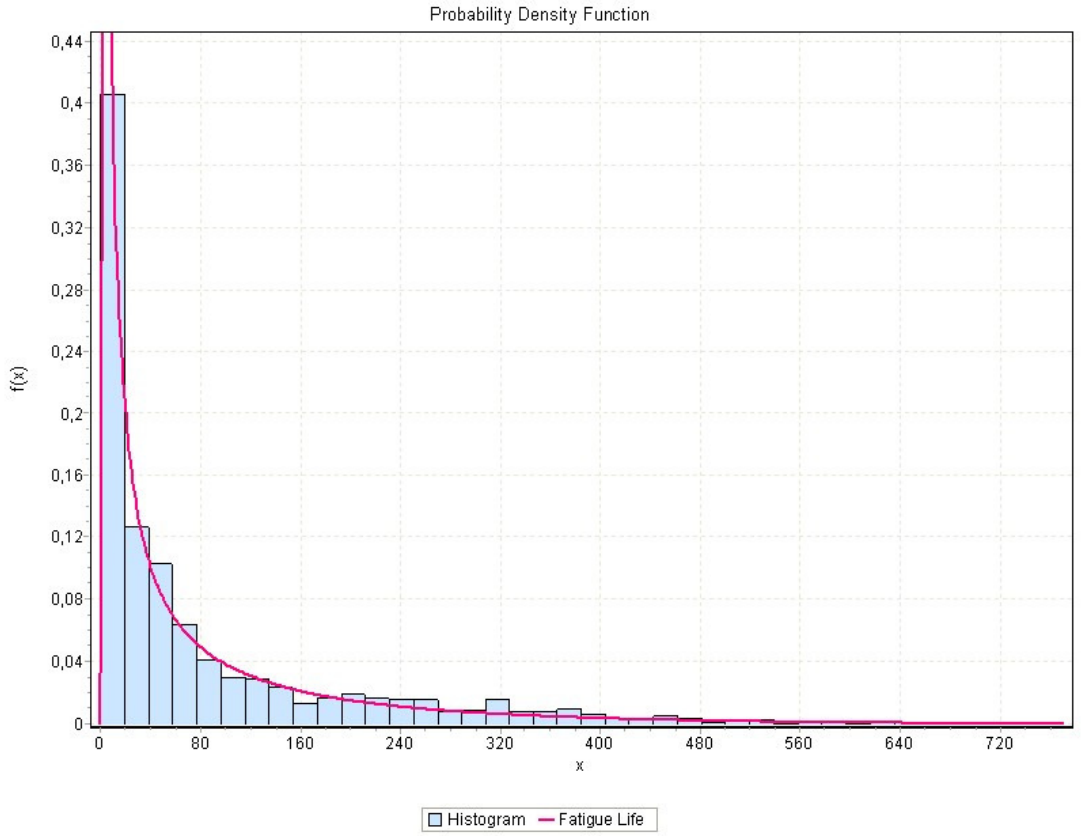

Figura 64 – FDP para *flags* SYN isoladas no ambiente *port scan* N para N modo *normal*

O número de *flags* SYN isoladas por segundo para o **modo *sneaky*** do ambiente *port scan* N para N apresenta um comportamento igual ao apresentado pelo ambiente *clean*. A

distribuição que melhor ajusta os dados é do tipo Johson SB, como ilustrado na figura 65.

A maior percentagem de *flags* SYN isoladas por segundo é obtida para valores bastante baixos, entre 3 e 5, e o número máximo de *flags* SYN isoladas por segundo não ultrapassa as 15.

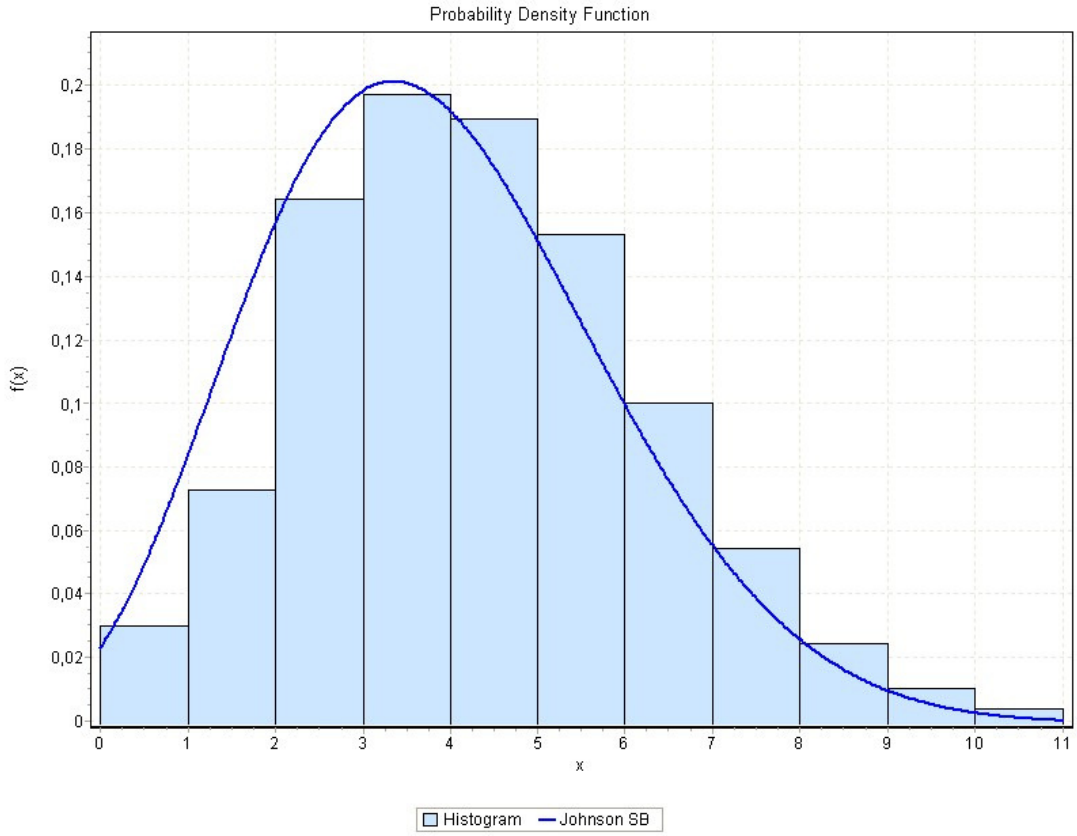

Figura 65 – FDP para *flags* SYN isoladas no ambiente *port scan* N para N modo *sneaky*

A análise, nos dois modos, para o número de *flags* SYN não isoladas por segundo é semelhante à realizada para o número de *flags* SYN isoladas por segundo.

O **modo *normal*** apresenta uma distribuição do tipo *Fatique Life*, com mais de 50% das contagens nos primeiros três intervalos do histograma e com contagens de *flags* SYN não isoladas por segundo bastante elevadas, superiores a 700. Na figura 66 está

ilustrado um exemplo representativo dos gráficos das funções densidade de probabilidade obtidas para este modo.

O **modo *sneaky***, representado na figura 67, apresenta tal como no ambiente *clean* um baixo número de *flags* SYN não isoladas por segundo, sendo mesmo assim aproximadamente o dobro das *flags* SYN isoladas por segundo no mesmo modo.

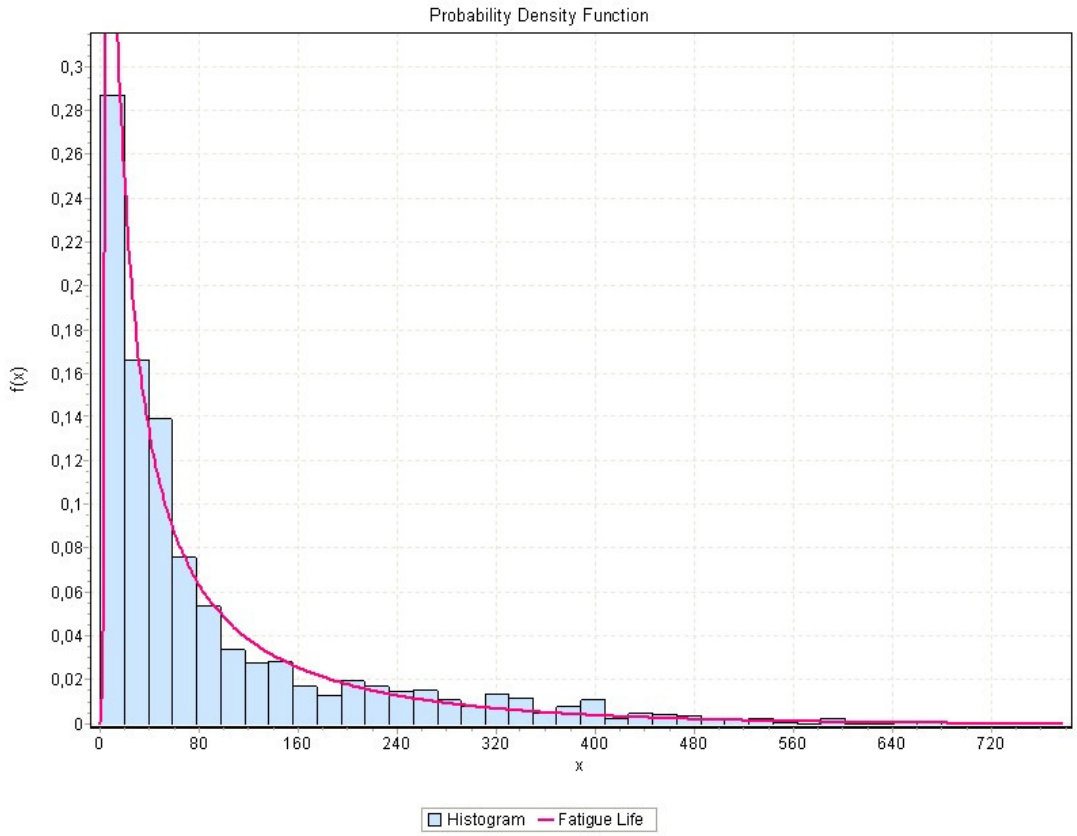

Figura 66 – FDP para *flags* SYN não isoladas no ambiente *port scan* N para N modo *normal*

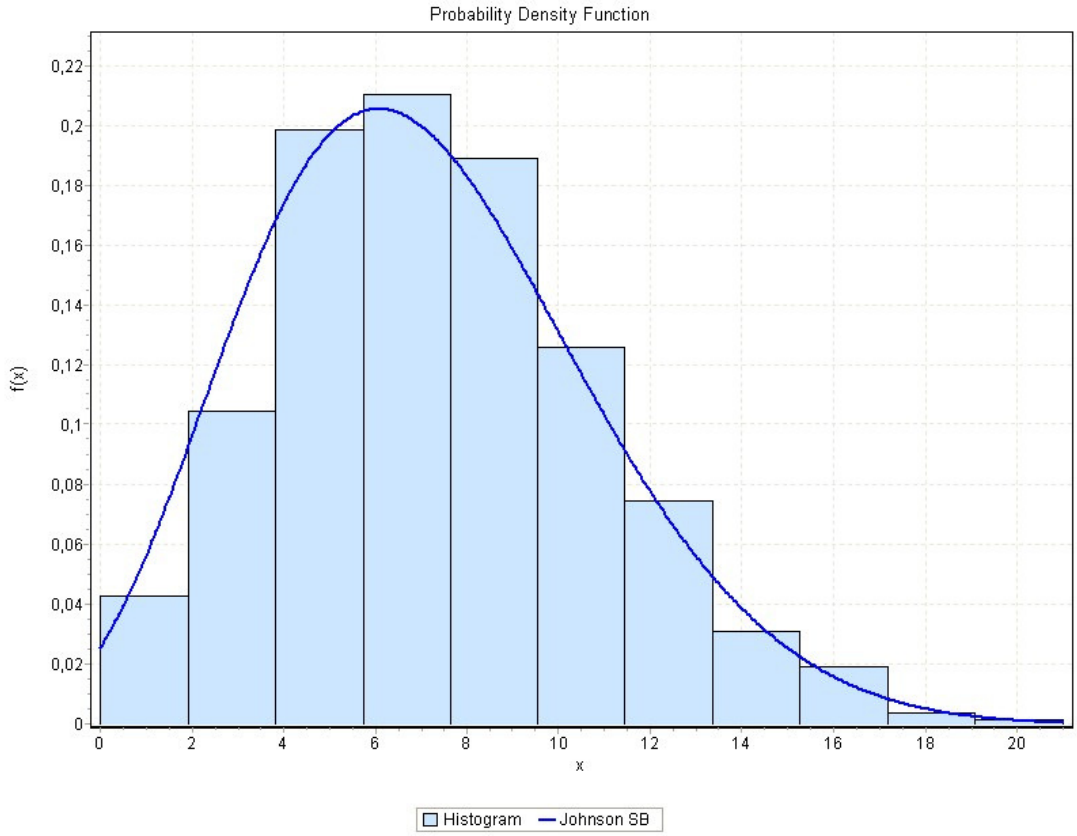

Figura 67 – FDP para *flags* SYN não isoladas no ambiente *port scan* N para N modo *sneaky*

Como o modo *sneaky* do ambiente *port scan* N para N e o ambiente *clean* têm comportamentos iguais na distribuição de *flags* SYN isoladas e não isoladas por segundo, passou-se para a análise da razão entre o número de *flags* SYN não isoladas e isoladas por segundo, resultados que são apresentados na tabela 9.

Como se pode observar, a razão entre ambas apresenta um valor sempre inferior a 2, tendo 11 dos 12 casos razões inferiores a 1.88, enquanto que no ambiente *clean* esta razão era sempre maior ou igual a 2.

Número total de *flags* SYN não isoladas.	**Número total de *flags* SYN isoladas.**	**Razão entre número de *flags* SYN não isoladas e isoladas**
11240	5996	1,87458305537025
12982	7034	1,84560705146432
12597	6838	1,84220532319392
12707	6892	1,84373186302960
12512	6797	1,84081212299544
12774	6927	1,84408834993504
12682	6884	1,84224288204532
12682	6880	1,84331395348837
12732	6905	1,84388124547429
13572	7043	1,92701973590799
12608	6841	1,84300540856600
12681	6882	1,84263295553618
Média		
12647,4166666667	6826,58333333333	1,85267154140065

Tabela 9 – Razão entre número de *flags* SYN não isoladas e isoladas para o ambiente *port scan* N para N modo *sneaky*

5.5.2 – Intervalo de tempo entre pacotes

Devido à limitação imposta pelo EasyFit, como explicado no ponto 5.1.2, apenas os primeiros 250 000 valores do vector de dados são passados para o programa. Por uma questão de tempo de processamento, dos 250 000 valores apenas os 50 000 valores contidos no intervalo [150 000 – 200 000] são considerados.

Tal como nos restantes ambientes, os modos ***normal*** e ***sneaky*** para o **ambiente *port scan* 1 para N** apresentam como distribuição que melhor ajusta os dados obtidos uma distribuição do tipo *Fatique Life.*

Nas figuras 68 e 69 estão representados dois dos gráficos da função densidade de probabilidade obtidos para os modos *normal* e *sneaky*, respectivamente. É possível observar que a quase totalidade dos pacotes chega com um intervalo de tempo entre si inferior a 0.005 segundos.

Na tabela 5 estão registadas as médias dos intervalos de tempo entre pacotes dos dois casos apresentados nas figuras 68 e 69

	Média do intervalo de tempo entre pacotes
Sneaky	0,00107
Normal	0,00116

Tabela 10 – Média do intervalo de tempo entre pacotes no ambiente *port scan* 1 para N

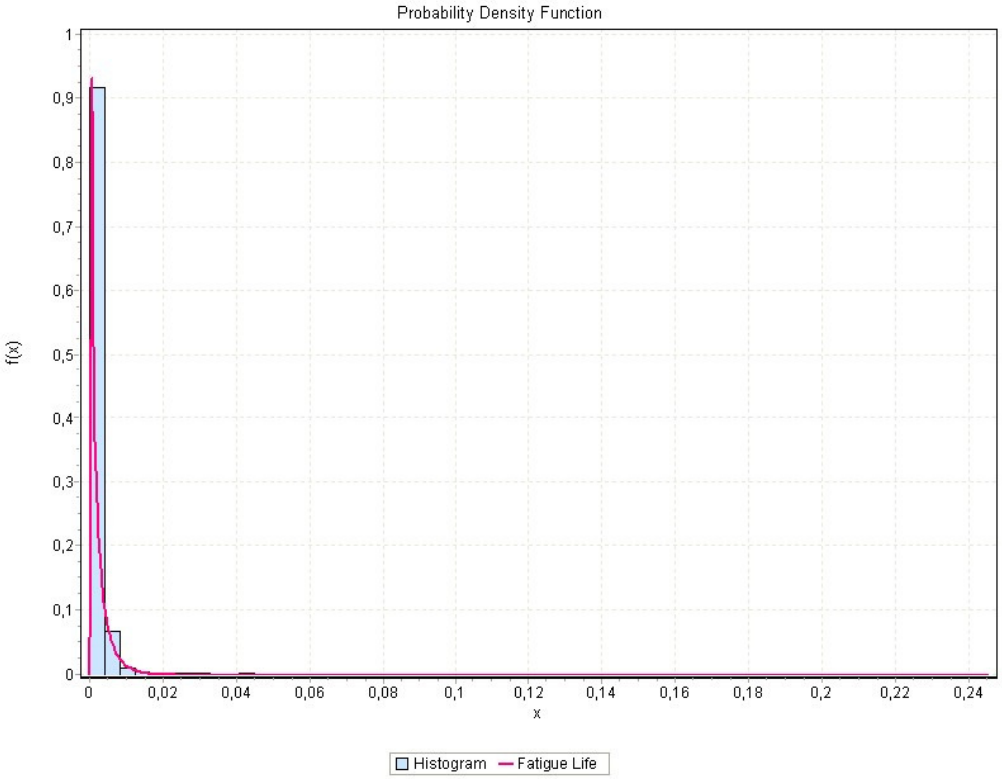

Figura 68 – FDP para o intervalo de tempo entre pacotes no ambiente *port scan* N para N modo *normal*

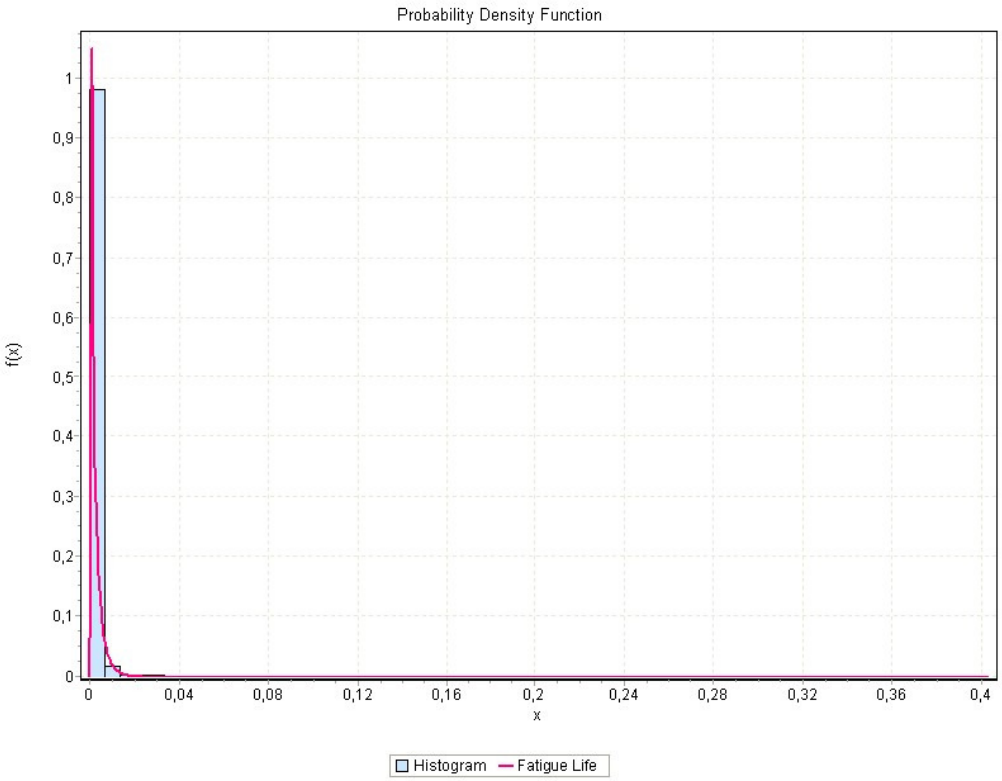

Figura 69 – FDP para o intervalo de tempo entre pacotes no ambiente *port scan* N para N modo *sneaky*

5.5.3 – Portos TCP quando só as *flags* SYN se encontram activas

A função de distribuição do tipo *Johson SB* é a que melhor ajusta a função densidade de probabilidade dos portos de origem quando só as *flags* SYN estão activas no **modo *normal*** do ambiente *port scan* N para N, tal como é ilustrado no exemplo da figura 70.

A gama de portos de origem está compreendida entre, aproximadamente, 32000 e 65000, apresentado percentagens mais elevadas de ocorrências para alguns intervalos de portos, percentagens que variam entre os 2,8% e menos de 0,2% para alguns intervalos, havendo diversos intervalos com percentagens intermédias na casa do 1,4%.

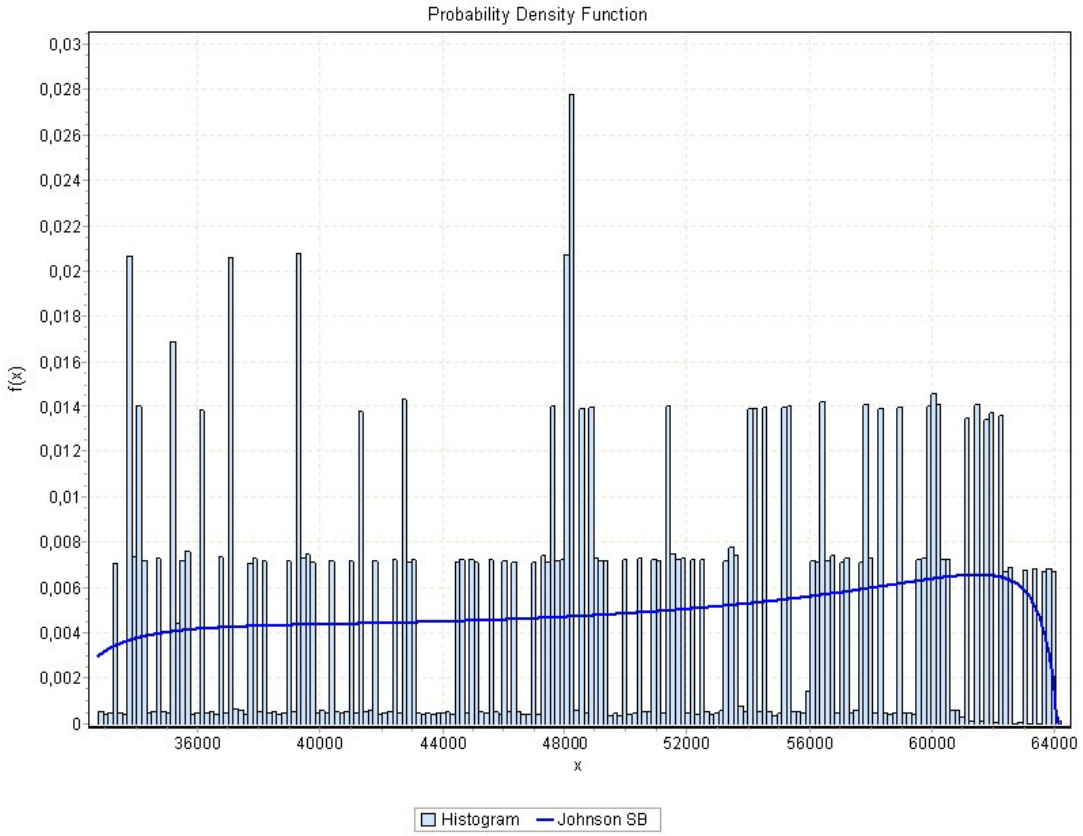

Figura 70 – FDP para os portos de origem no ambiente *port scan* N para N modo *normal*

O **modo *sneaky*** apresenta uma distribuição do tipo *Wakeby,* em que grande parte dos intervalos de portos de origem ocorre com percentagens de cerca de 0,5%, havendo diversos picos de percentagens em alguns intervalos, alguns superiores a 3,5%, desfazendo assim o comportamento do ambiente *clean*. Os portos de origem têm todos valores superiores a aproximadamente 32000. Na figura 71 está ilustrado um dos gráficos obtidos para a FDP obtida.

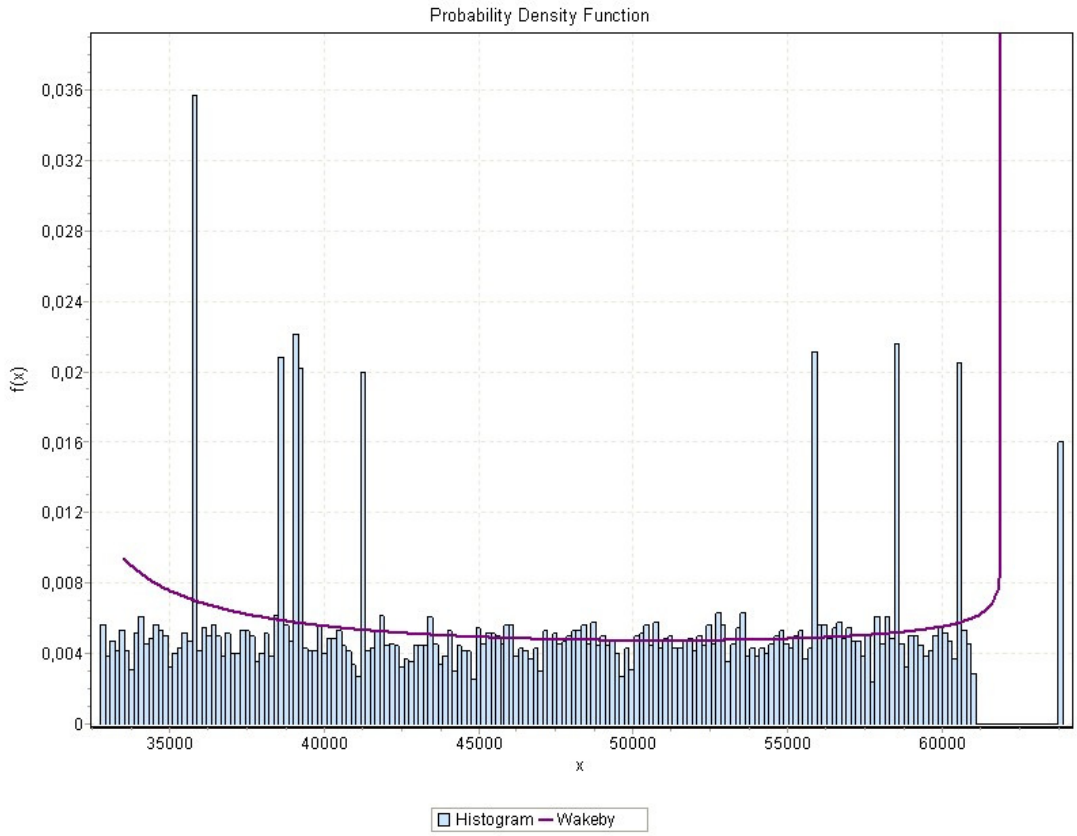

Figura 71 – FDP para os portos de origem no ambiente *port scan* N para N modo *sneaky*

Os portos de destino quando só as *flags* SYN estão activas apresentam, no **modo *normal*** do ambiente *port scan* N para N, uma distribuição do tipo *Pareto* 2, como se pode ver na figura 72.

À semelhança dos outros ambientes de *port scan,* os portos de destino que mais vezes surgem quando só as *flags* SYN estão activas

têm valores inferiores a 10000, sendo que mais de 10% dos portos de destino têm número entre 1 e aproximadamente 320. Com excepção dos picos de cerca de 2% para intervalos próximos dos valores 33000 e 49000, os restantes intervalos de portos de destino apresentam percentagens quase nulas.

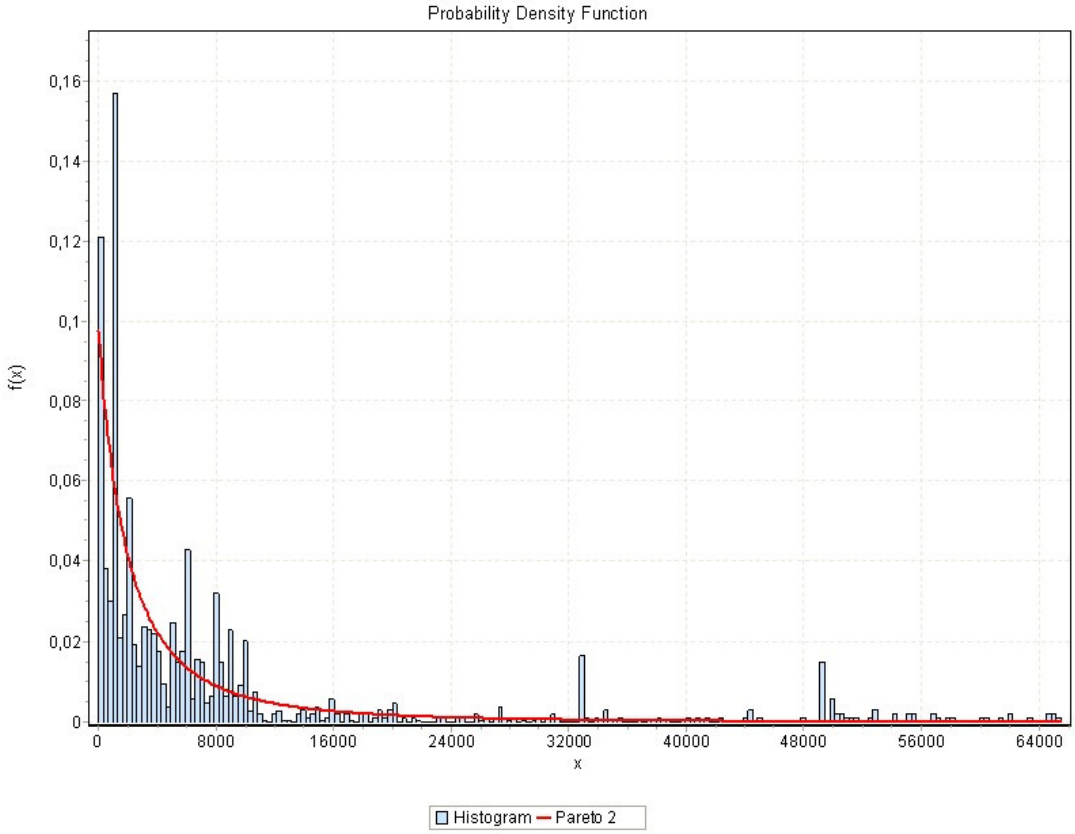

Figura 72 – FDP para os portos de destino no ambiente *port scan* N para N modo *normal*

Os portos de destino quando só as *flags* SYN estão activas no **ambiente de *port scan* N para N e modo *sneaky*,** apresentam uma distribuição do tipo *Burr, tal* como ilustrado na figura 73

Apesar de ser uma distribuição de tipo diferente da apresentada pelo ambiente *clean,* são visíveis as semelhanças entre ambos os ambientes. A grande maioria dos portos de destino correspondem aos portos mais comuns, como explicado para o ambiente *clean*, ou seja, têm valores entre 1 e aproximadamente 330, sendo a percentagem

de incidência para os outros intervalos de portos, na totalidade da gama de portos, nula ou quase nula.

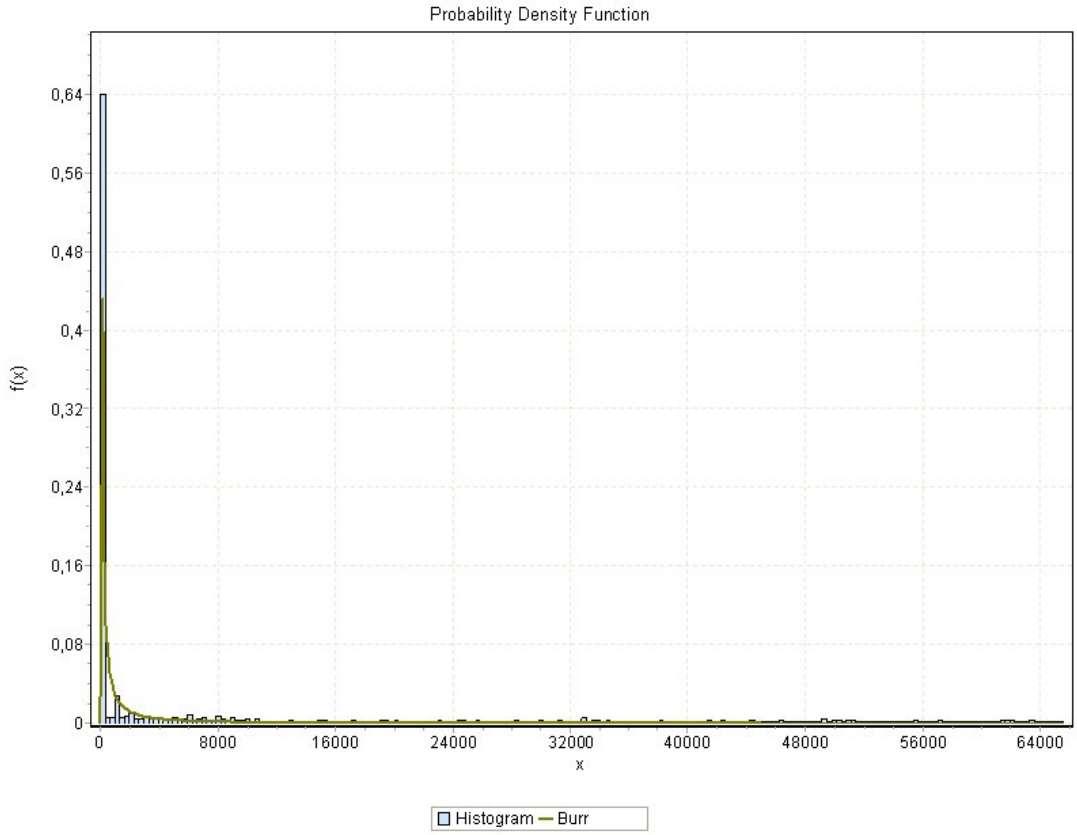

Figura 73 – FDP para os portos de destino no ambiente *port scan* N para N modo *sneaky*

6 – Conclusões

As conclusões deste trabalho assentam nas comparações efectuadas entre o padrão correspondente ao ambiente *clean* e os padrões dos restantes ambientes e são apresentadas para cada um dos tipos de dados obtidos.

6.1 - Número de *flags* SYN por segundo

Na tabela 11 estão reunidos todos os tipos de distribuição que melhor ajustaram o número de *flags* SYN isoladas e não isoladas por segundo para cada modo de cada ambiente analisado.

		Aggressive	*Normal*	*Sneaky*
***Flags* SYN isoladas**	Clean	*Johson SB*		
	Port scan 1 para 1	*Log Logistic*	*Log Logistic*	*Johson SB*
	Port scan 1 para N	Não aplicável	*Burr*	*Johson SB*
	Port scan N para 1	*Burr*	*Wakeby*	*Burr*
	Port scan N para N	Não aplicável	*Fatique Life*	*Johson SB*
***Flags* SYN não isoladas**	Clean	*Johson SB*		
	Port scan 1 para 1	*Log Logistic*	*Log Logistic*	*Johson SB*
	Port scan 1 para N	Não aplicável	*Burr*	*Johson SB*
	Port scan N para 1	*Burr*	*Lognormal*	*Burr*
	Port scan N para N	Não aplicável	*Fatique Life*	*Johson SB*

Tabela 11 – Distribuições das FDP para o número de *flags* SYN por segundo

Os **modos, *aggressive* e *normal*** de todos os ambientes e o **modo *sneaky* do ambiente *port scan* N para** 1, apresentam

distribuições para a FDP diferentes do ambiente *clean* e comportamentos também distintos, nomeadamente no número máximo de *flags* SYN contadas por segundo. Nestes modos, o número máximo de *flags* SYN por segundo é muito mais elevado que no ambiente clean (no ambiente clean temos máximos próximos de 20 e nos restantes modos falamos de máximos que chegam perto de 1000), indicando um elevado número de pedidos de sessão TCP e indicando um comportamento anómalo face ao que é tido como *normal*. Pode-se assim concluir acerca da existência de *port scan* na rede, e fazendo uso da distribuição que melhor ajusta os dados, pode-se ter uma ideia do tipo de *port scan* que está a ser efectuado. Em todos os casos analisados para os modos *aggressive* e *normal* e em todos os ambientes e no modo *sneaky* em ambiente *port scan* N para 1, foi possível detectar um comportamento, a nível do número de *flags* SYN isoladas e não isoladas, bastante distinto do ambiente clean, tendo sido obtida uma taxa de 100% na detecção de *port scan*s.

Comparando as distribuições do ambiente *clean* com as distribuições dos restantes modos de cada ambiente é possível ver que apenas os **modos *sneaky***, com excepção do modo *sneaky* em ambiente *port scan* N para 1, apresentam a mesma distribuição Johson SB. Para além de apresentarem a mesma distribuição, apresentam comportamentos muito semelhantes na forma como o número de contagens de *flags* SYN isoladas e não isoladas por segundo se distribui nos histogramas. Assim, olhando só para estes dados não é seguro nem conclusivo afirmar a existência de qualquer anomalia.

A análise, em MatLab, da razão entre o número de *flags* SYN não isoladas e isoladas apresenta já resultados que permitem diferenciar o ambiente clean dos restantes modos *sneaky* referidos. Enquanto que o ambiente clean apresenta uma razão nunca inferior a 2, os restantes modos apresentam uma razão nunca superior ou igual

a 2. De todas as razões calculadas para os modos *sneaky* (36, 12 para cada ambiente), apenas uma delas, no modo *sneaky* do ambiente *port scan* 1 para N, deu um valor superior a 2. Pode-se, perante estes resultados, concluir com elevado grau de certeza a existência de *port scan* na rede, não sendo no entanto possível uma distinção clara entre diferentes tipos de *port scan*, uma vez que os ambientes de *port scan* N para N e 1 para N apresentam razões praticamente iguais, tendo apenas o ambiente *port scan* 1 para 1 apresentado uma razão de valor superior.

6.2 – Intervalo de tempo entre pacotes

Da análise do intervalo de tempo entre pacotes nos vários ambientes não foi possível retirar conclusões acerca da detecção de *port scans,* pois todos os ambientes e todos os modos apresentam uma distribuição do tipo *Fatique Life,* como é visível na tabela 12, e as mesmas características nos histogramas. A análise das médias dos tempos entre pacotes também não foi conclusiva, pois para todos os ambientes e modos as médias são muito próximas.

		Aggressive	*Normal*	*Sneaky*
Intervalo de tempo entre pacotes	Clean	*Fatique Life*		
	Port scan 1 para 1	*Fatique Life*	*Fatique Life*	*Fatique Life*
	Port scan 1 para N	*Fatique Life*	*Fatique Life*	*Fatique Life*
	Port scan N para 1	*Fatique Life*	*Fatique Life*	*Fatique Life*
	Port scan N para N	*Fatique Life*	*Fatique Life*	*Fatique Life*

Tabela 12 – Distribuições das FDP para o intervalo de tempo entre pacotes

6.3 – Portos TCP quando só as *flags* SYN se encontram activas

A análise dos portos TCP quando só as *flags* SYN se encontravam activas gerou diversos tipos de distribuição para as FDP, tal como se pode ver na tabela 13. Apesar de alguns modos em ambiente *port scan* apresentarem a mesma distribuição que o ambiente clean, nem sempre apresentam comportamento igual.

A análise dos portos de origem permite ver, apesar de alguma igualdade no tipo de distribuição da FDP, que todos os modos dos quatro ambientes de *port scan* estudados apresentam diferenças acentuadas em relação ao ambiente *clean*. Enquanto que o ambiente *clean* tem mais ou menos a mesma percentagem para cada um dos intervalos da gama de portos de origem, os restantes modos apresentam, em todos os casos analisados, diversas gamas de portos de origem com percentagens bastante elevadas, quando comparadas com as restantes. Mesmo o modo *sneaky* apresenta um ou mais destes picos em todos os casos estudados, diferenciando-se do ambiente *clean*. É portanto possível a detecção dos *port scan* nos diferentes modos e ambientes, sendo também possível uma detecção do modo conhecendo o ambiente de *port scan* ou detecção do ambiente conhecendo o modo, uma vez que diferentes modos de diferentes ambientes *port scan* podem ser facilmente confundidos. Através da análise das FDP e do número de vezes que cada intervalo dos portos de origem surge foi possível uma taxa de sucesso de 100% na detecção dos *port scan*.

		Aggressive	*Normal*	*Sneaky*
Portos de origem	Clean	*Wakeby*		
	Port scan 1 para 1	*Wakeby*	*Wakeby*	*Wakeby*
	Port scan 1 para N	Não aplicável	*Wakeby*	*Wakeby*
	Port scan N para 1	*Wakeby*	*Wakeby*	*Wakeby*
	Port scan N para N	Não aplicável	*Johnson SB*	*Wakeby*
Portos de destino	Clean	*Phased Bi-Exponential*		
	Port scan 1 para 1	*Phased Bi-Weibull*	*Fatique Life*	*Phased Bi-Exponential*
	Port scan 1 para N	Não aplicável	*Dagum*	*Phased Bi-Weibull*
	Port scan N para 1	*Person 6*	*Pareto 2*	*Gamma*
	Port scan N para N	Não aplicável	*Pareto 2*	*Burr*

Tabela 13 – Distribuições das FDP para os portos TCP quando só as *flags* SYN estão activas

Os portos de destino quando só as *flags* SYN estão activas também permitem uma detecção dos ataques de *port scan* através da análise das funções densidade de probabilidade.

Apenas o modo *sneaky* do ambiente *port scan* 1 para 1 apresenta o mesmo tipo de distribuição que o ambiente *clean*, uma distribuição tipo *Phased Bi-Exponential*. Olhando para ambos os casos, verifica-se a semelhança existente entre eles, tornando impossível retirar qualquer conclusão quanto à existência de um ataque ou não. Nos restantes modos dos restantes ambientes a existência de diferenças para o ambiente *clean* é mais ou menos evidente, consoante o caso. O ambiente *clean* apresenta um grande número de portos de destino no intervalo de 1 a 320, sendo quase nulas e de idêntico valor as contagens para os restantes intervalos. Os restantes modos nos diferentes ambientes apresentam igualmente

elevado número de portos de destino com valores entre 1 e 320; contudo, com excepção do já referido modo *sneaky* em *port scan* 1 para 1, apresentam depois valores superiores aos apresentados no modo *clean* para outros intervalos de portos, nomeadamente nos portos com número inferior a 10000 e em redor dos números 33000 e 49000, aproximadamente. Das 10 combinações de modos e ambientes existentes neste trabalho é possível a detecção, por comparação com o ambiente *clean,* em 9 dessas combinações.

7 - Trabalho Futuro

De forma a completar o trabalho apresentado nesta dissertação, existem alguns tópicos de interesse que podem ser abordados e desenvolvidos no futuro:

- Análise de diferentes tipos de ataques à segurança das redes, nomeadamente outros tipos de *port scan*s;
- Expandir o número de variáveis estudadas para a detecção dos ataques, nomeadamente nos casos de *port scan* em modo *sneaky*;
- Reduzir o tempo das capturas de forma a tornar mais rápida a detecção;
- Uso de novas ferramentas estatísticas que disponham de outras funcionalidades de forma a alargar o leque de opções para a detecção de ataques;
- Criação de uma plataforma que automatize as operações de análise e detecção, de forma a facilitar a tarefa dos gestores de rede.

Bibliografia

[1] A. Fadia and M. Zacharia, *Network intrusion alert: an ethical hacking guide to intrusion detection*: Thomson Course Technology PTR, 2007.

[2] CERT. *Statistics*. Available: http://www.cert.org/stats

[3] R. G. Bace, *Intrusion detection systems [electronic resource] / Rebecca Bace and Peter Mell*. [Gaithersburg, MD :: U.S. Dept. of Commerce, Technology Administration, National Institute of Standards and Technology, 2001.

[4] J. P. Anderson, "Computer Security Threat Monitoring and Surveillance," 1980.

[5] A. Adelsbach*, et al.*, "Conceptual Model and Architecture of MAFTIA," Faculdade de Ciências da Universidade de Lisboa, Lisboa, Fevereiro de-2003

[6] A. Andress, *Surviving Security: How to Integrate People, Process, and Technology, Second Edition*: Taylor and Francis, 2003.

[7] V. Kumar*, et al.*, *Managing cyber threats: issues, approaches, and challenges*: Springer, 2005.

[8] CERT. *IP Denial-of-Service Attacks*. Available: http://www.cert.org/advisories/CA-1997-28.html

[9] ZDNET. Available: http://www.zdnet.com/blog/security/windows-7-vista-exposed-to-teardrop-attack/4222

[10] Microsoft. Available: http://www.microsoft.com/technet/security/advisory/975497.mspx

[11] N. S. University. *Ethics in Computing*. Available: http://ethics.csc.ncsu.edu/abuse/wvt/worm/

[12] R. Anderson, *Security engineering: a guide to building dependable distributed systems*: Wiley, 2008.

[13] CNN. Available: http://archives.cnn.com/2000/TECH/computing/03/01/prevent.ddos.idg/index.html

[14] F. Wilder, *A Guide to theTVP/IP Protocol Suite*. London: Arthech House, 1993.

[15] C. L. Schuba*, et al.*, "Analysis of a denial of service attack on TCP," in *Security and Privacy, 1997. Proceedings., 1997 IEEE Symposium on*, 1997, pp. 208-223.

[16] M. Handley, "Internet Denial-of-Service Considerations," 2006.

[17] O. Arkin, "Network Scanning Techniques," 1999.

[18] CERT. *Microsoft Outlook and Outlook Express Cache Bypass Vulnerability*. Available: http://www.cert.org/advisories/CA-2000-14.html

[19] Microsoft. *Microsoft Portugal*. Available: http://www.microsoft.com/portugal/athome/security/viruses/virus101.mspx

[20] M. Szczesiak, "Computer Viruses," ed. University of Bristol, 2004.

[21] *Computer Knowledge*. Available: http://www.cknow.com/cms/vtutor/number-of-viruses.html

[22] Symantec, "Symantec Global Internet Security Threat Report Trends for 2009," 2009.

[23] BBC. *News*. Available: http://news.bbc.co.uk/2/hi/technology/2693925.stm

[24] Slashdot.org. Available: http://slashdot.org/article.pl?sid=03/01/25/1245206&mode=flat&tid=109

[25] CERT. *Love Letter Worm*. Available: http://www.cert.org/advisories/CA-2000-04.html

[26] Scripting.com. Available: http://scripting.com/davenet/2000/11/15/clayshirkyonp2p.html

[27] D. E. Denning, "An Intrusion-Detection Model," *Software Engineering, IEEE Transactions on,* vol. SE-13, pp. 222-232, 1987.

[28] J. M. Estevez-Tapiador*, et al.*, "Anomaly detection methods in wired networks: a survey and taxonomy," *Computer Communications,* vol. 27, pp. 1569-1584, 2004.

[29] H. Debar*, et al.*, "Towards a taxonomy of intrusion-detection systems," *Computer Networks,* vol. 31, pp. 805-822, 1999.

[30] S. AXELSSON, "Research in Intrusion-Detection Systems: A Survey. Technical Report TR-98-17," Chalmers University Technology, Göteborg, Sweden, 1999.

[31] P. G. Neumann and P. A. Porras, "Experience with EMERALD to Date," presented at the Proceedings of the Workshop on Intrusion Detection and Network Monitoring, 1999.

[32] F. Desheng*, et al.*, "Research on a Distributed Network Intrusion Detection System Based on Association Rule Mining," in *Information Science and Engineering (ICISE), 2009 1st International Conference on*, 2009, pp. 1816-1818.

[33] R. S. NETTO, "Detecção de Intrusão Utilizando Redes Neurais Artificiais no Reconhecimento de Padrões de Ataque," Universidade Federal de Itajubá, Itajubá, 2006.

[34] SNORT. Available: http:www.snort.org

[35] Haystack. Available: http://www.haystacknetwork.com/

[36] S. E. Smaha, "Haystack: an intrusion detection system," in *Aerospace Computer Security Applications Conference, 1988., Fourth*, 1988, pp. 37-44.

[37] A. Sundaram, "An introduction to intrusion detection," *Crossroads,* vol. 2, pp. 3-7, 1996.

[38] M. H. P. C. Chaves, "Análise de estado de tráfego de redes TCP/IP para aplicação em detecção de intrusão," Dissertação

(Mestrado em Computação Aplicada), Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2002.

[39] T. F. Lunt*, et al.*, "A REAL-TIME INTRUSION-DETECTION EXPERT SYSTEM (IDES), SRI Tecnhical report," 1992.

[40] D. Anderson*, et al.*, "Next-generation Intrusion Detection Expert System (NIDES) A Summary, SRI Tecnhical report," 1995.

[41] D. Farmer, "The COPS security checker system , Technical Report CSD-TR-993," Purdue University1994.

[42] D. R. Safford*, et al.*, "The TAMU security package: an ongoing response to internet intruders in an academic environment," presented at the Proceedings of the 4th conference on UNIX security symposium - Volume 4, Santa Clara, California, 1993.

[43] E. H. Spafford and D. Zamboni, "Intrusion detection using autonomous agents," *Comput. Netw.,* vol. 34, pp. 547-570, 2000.

[44] C. M. D. S. Miranda, "IMPLEMENTAÇÃO DE CENARIOS PARA OBTENÇÃO DE CONHECIMENTO ABSOLUTO DE REDE " Master Mestrado, Departamento de Electrónica, Telecomunicações e Informática, Universidade de Aveiro, Aveiro, 2009.

[45] GNS3. Available: http://www.gns3.net/

[46] Nmap. Available: http://nmap.org/

[47] Tshark. Available: http://www.wireshark.org/docs/man-pages/tshark.html

[48] Wireshark. Available: http://www.wireshark.org/

[49] Mathwave. *EasyFit*. Available: http://www.mathwave.com/easyfit-distribution-fitting.html

[50] MATLAB. Available: http://www.mathworks.com/products/matlab/

[51] IANA. (2010). *PORT NUMBERS*. Available: http://www.iana.org/assignments/port-numbers

[52] AWK. *AWK man page*. Available:
http://unixhelp.ed.ac.uk/CGI/man-cgi?awk

[53] GNU. *SHELL*. Available:
http://www.gnu.org/software/bash/manual/bashref.html

Anexos

Anexo A – Shell Scripts

Shell é um interpretador de comandos, a funcionar no sistema operativo Linux, usado como meio de interpretação entre o utilizador e o computador. Existem várias implementações de Shell, sendo a mais comum o bash.

O Bash é o shell Unix, escrito por Brian Fox, desenvolvido para o projecto GNU, da Free Software Foundation, que se tornou padrão nas várias distribuições Linux. O nome Bash é um acrónimo para Bourne Again Shell, um trocadilho para com Sthephen Bourne, autor do ancestral directo do actual shell Unix sh.[53]

Os comandos podem ser enviados para serem interpretados de duas formas distintas:

- Interactiva: os comandos são inseridos na linha de comandos e passados um a um para o interpretador.

- Não interactiva: os comandos são passados para o interpretador através de um ficheiro shell script. Estes scripts são ficheiros de texto contendo sequências de comandos para serem executados.

A primeira linha de um script indica o nome do shell que vai interpretar o script. Nesta dissertação usou-se o interpretador bash, assim a primeira linha de instruções é **#!/bin/bash.**

Para a execução dos sripts podem ser usados os seguintes comandos:

Comando	Condição
./nome_script	Se o script se encontra no directório actual.
$caminho/nome_script	Se o script não se encontra no directório actual

Tabela 14 – Comandos para execução de um script

Para a obtenção dos diversos dados foram usados os seguintes scripts:

```
#!/bin/bash

ls > nome.txt
for i in `cat nome.txt`
do
j=`echo $i | cut -d"." -f1 | sed 's/$/.txt/'`

        echo `tshark -r $i -qz io,stat,1, "COUNT(tcp)tcp[13] &
2,COUNT(tcp)tcp[13]==2" > /caminho_destino/$j`
        echo "$j done"
        sleep 1
done
```

Figura 74 – Bash script *flags*SYN.sh

```
#!/bin/bash

ls > nome1.txt

j=-1
for i in `nome1.txt`
do
h1=nome2_$j
h=`echo $h1 | sed 's/$/.txt/'`

	echo `tshark -r $i -T fields -e frame.time_delta >
/caminho_destino/$h`
	echo "$h done"
	j=`expr $j + 1`
	echo $h
	sleep 1
done
```

Figura 75 – Bash script frame_time_delta.sh

```
#!/bin/bash

ls > nome1.txt

j=-1
for i in `nome1.txt`
do
h1=nome2_$j
h2=nome3_$j
h=`echo $h1 | sed 's/$/.txt/'`
l=`echo $h2 | sed 's/$/.txt/'`

	echo `tshark -r $i -n -z io,stat,0 "tcp[13]==2" > /caminho/$h`
	echo "$h done"
	echo `awk '{print $7;}' /caminho_origem/$h >
/caminho_destino/$l`
	echo "$l done"
	j=`expr $j + 1`
	echo $h
	echo $l
	sleep 1
done
```

Figura 76 – Bash script TCP_Source_PORT.sh

```
#!/bin/bash

ls > nome1.txt

j=-1
for i in `nome1.txt`
do
h1=nome2_$j
h2=nome3_$j
h=`echo $h1 | sed 's/$/.txt/'`
l=`echo $h2 | sed 's/$/.txt/'`

        echo `tshark -r $i -n -z io,stat,0 "tcp[13]==2" > /caminho/$h`
        echo "$h done"
        echo `awk '{print $9;}' /caminho_origem/$h >
/caminho_destino/$l`
        echo "$l done"
        j=`expr $j + 1`
        echo $h
        echo $l
        sleep 1
done
```

Figura 77 – Bash script TCP_Destination_PORT.sh

Anexo B – M-files

M-files usados em Matlab para obtenção de forma automática da razão entre *flags* SYN não isoladas e *flags* SYN isoladas

```
function importfile(fileToRead1) %ficheiros de saida -> data e text%

soma_syn(12,3)=0; %cria a matrix para os valores finais%
format short
for j=1:12  %ciclo para os 12 ficheiros de dados%
   eval(['soma_syn(j,1)=sum(data' num2str(j) '(:,1));']);
   eval(['soma_syn(j,2)=sum(data' num2str(j) '(:,2));']);
   soma_syn(j,3)=soma_syn(j,1)/soma_syn(j,2);
end;
```

Figura 78 – M-file para obtenção de dados estatísticos acerca das *flags* SYN por segundo

```
function importfile(fileToRead1)
%IMPORTFILE(FILETOREAD1)
%  Imports data from the specified file
%  FILETOREAD1:  file to read
%  Auto-generated by MATLAB

% Import the file
newData1 = importdata(fileToRead1);

% Create new variables in the base workspace from those fields.
vars = fieldnames(newData1);
for i = 1:length(vars)
   assignin('base', vars{i} newData1.(vars{i}));
end
```

Figura 79 – M-file importefile.m

Anexo C – Funções de Densidade de Probabilidade

Anexo C1 – Johson SB

A distribuição do tipo *Johnson-SB* apresenta a seguinte função densidade de probabilidade:

$$f(x)=\frac{\delta}{\lambda\sqrt{2\pi}z(1-z)}\exp\left(-\frac{1}{2}\left(\gamma+\delta\ln\left(\frac{z}{1-z}\right)\right)^2\right)$$

onde:

$z \equiv \frac{x-\xi}{\lambda}$

sendo os seus parâmetros:

γ - Continuous shape parameter

δ - Continuous shape parameter (δ >0)

λ - Continuous scale parameter (λ >0)

ξ - Continuous location parameter

Domínio:

$\zeta \le x \le \zeta + \lambda$

Anexo C2 – *Fatique Life*

A distribuição do tipo *Fatique Life* apresenta a seguinte função densidade de probabilidade:

$$f(x)=\frac{\sqrt{(x-\gamma)/\beta}+\sqrt{\beta/(x-\gamma)}}{2\alpha(x-\gamma)}\times\phi\left(\frac{1}{\alpha}\left(\sqrt{\frac{x-\gamma}{\beta}}-\sqrt{\frac{\beta}{x-\gamma}}\right)\right)$$

em que ϕ é a função densidade de probabilidade da distribuição *normal* padrão igual a:

$$\phi(x)=\frac{e^{-x^2/2}}{\sqrt{2\pi}}$$

com parâmetros:

α - Continuous shape parameter (α > 0)

β - Continuous scale parameter (β > 0)

γ - Continuous location parameter

Domínio:

γ < x < $+\infty$

Anexo C3 – *Wakeby*

A distribuição do tipo *Wakeby* apresenta a seguinte função densidade de probabilidade:

$$x(F)=\zeta+\frac{\alpha}{\beta}\left(1-(1-F)^{\beta}\right)-\frac{\gamma}{\delta}\left(1-(1-F)^{-\delta}\right)$$

onde F é uma variável aleatória uniforme padrão. Ou seja, a equação acima define a função de ponto percentual para a distribuição *Wakeby*, onde $F=F(x)=P(X\le x)$.

Os parâmetros (todos contínuos) são:

β, γ, δ - Shape parameter;

ζ , α - Location parameter.

sendo impostas as seguintes condições:

$\alpha \neq 0 \quad ou \quad \gamma \neq 0;$
$\beta + \delta > 0 \quad ou \quad \beta = \gamma = \delta = 0;$
$se \quad \alpha = 0 \quad então \quad \beta = 0;$
$se \quad \gamma = 0 \quad então \quad \delta = 0;$
$\gamma \geq 0 \quad e \quad \alpha + \gamma \geq 0.$

Domínio:

$\zeta \leq x < \infty \quad se \quad \delta \geq 0 \quad e \quad \gamma > 0;$
$\zeta \leq x \leq \zeta + \alpha/\beta - \gamma/\delta \quad se \quad \delta < 0 \quad ou \quad \gamma = 0.$

Anexo C4 – *Phased Bi-Exponential*

A distribuição do *Phased Bi-Exponential* apresenta a seguinte função densidade de probabilidade:

$$f(x) = \begin{cases} \lambda_1 e^{-\lambda_1 (x-\gamma_1)} & \gamma_1 \leq x \leq \gamma_2 \\ \lambda_2 e^{-\lambda_2 (x-\gamma_2) - \lambda_1 (\gamma_2 - \gamma_1)} & \gamma_2 < x < +\infty \end{cases}$$

com parâmetros:

λ_1 - Continuous inverse scale parameter (λ_1 >0);

γ_1 - Continuous location parameter;

λ_2 - Continuous inverse scale parameter (λ_2 >0);

γ_2 - Continuous location parameter ($\gamma_2 > \gamma_1$).

Domínio:

$\gamma_1 \leq x < +\infty$

Anexo C5 – *Log-Logistic*

A distribuição do tipo *Log-Logistic* apresenta a seguinte função densidade de probabilidade:

$$f(x) = \frac{\alpha}{\beta}\left(\frac{x-\gamma}{\beta}\right)^{\alpha-1}\left(1+\left(\frac{x-\gamma}{\beta}\right)^{\alpha}\right)^{-2}$$

com parâmetros:

α - Continuous shape parameter (α > 0)

β - Continuous scale parameter (β > 0)

γ - Continuous location parameter

Domínio:

γ < x < $+\infty$

Anexo C6 – *Phased Bi-Weibull*

A distribuição do tipo *Phased Bi-Weibull* apresenta a seguinte função densidade de probabilidade:

$$f(x) = \begin{cases} \frac{\alpha_1}{\beta_1}\left(\frac{x-\gamma_1}{\beta_1}\right)^{\alpha_1-1}\exp\left(-\left(\frac{x-\gamma_1}{\beta_1}\right)^{\alpha_1}\right) & \gamma_1 \le x \le \gamma_2 \\ \frac{\alpha_2}{\beta_2}\left(\frac{x-\gamma_1}{\beta_2}\right)^{\alpha_2-1}\exp\left(-\left(\frac{x-\gamma_1}{\beta_2}\right)^{\alpha_2}\right) & \gamma_2 < x < +\infty \end{cases}$$

com parâmetros:

α_1 - Continuous shape parameter (α_1 > 0)

β_1 - Continuous scale parameter ($\beta_1 > 0$)

γ_1 - Continuous location parameter

α_2 - Continuous shape parameter ($\alpha_2 > 0$)

β_2 - Continuous scale parameter ($\beta_2 > 0$)

γ_2 - Continuous location parameter ($\gamma_2 > \gamma_1$)

Domínio:

$\gamma \leq x < +\infty$

Anexo C7 – *Burr*

A distribuição do tipo *Burr* apresenta a seguinte função densidade de probabilidade:

$$f(x) = \frac{\alpha k \left(\frac{x-\gamma}{\beta} \right)^{\alpha-1}}{\beta \left(1 + \left(\frac{x-\gamma}{\beta} \right)^{\alpha} \right)^{k+1}}$$

com parâmetros:

K – Continuous shape parameter (k> 0)

α - Continuous shape parameter ($\alpha > 0$)

β - Continuous scale parameter ($\beta > 0$)

γ - Continuous location parameter

Domínio:

$\gamma < x < +\infty$

Anexo C8 – *Dagum*

A distribuição do tipo *Dagum* apresenta a seguinte função densidade de probabilidade:

$$f(x) = \frac{\alpha k \left(\frac{x-\gamma}{\beta} \right)^{\alpha k - 1}}{\beta \left(1 + \left(\frac{x-\gamma}{\beta} \right)^{\alpha} \right)^{k+1}}$$

com parâmetros:

K – Continuous shape parameter (k> 0)

α - Continuous shape parameter (α > 0)

β - Continuous scale parameter (β > 0)

γ - Continuous location parameter

Domínio:

γ < x < $+\infty$

Anexo C9 – *Lognormal*

A distribuição do tipo *Lognormal* apresenta a seguinte função densidade de probabilidade:

$$f(x) = \frac{\exp\left(-\frac{1}{2} \left(\frac{\ln(x-\gamma) - \mu}{\sigma} \right)^2 \right)}{(x-\gamma)\sigma\sqrt{2\pi}}$$

com parâmetros:

σ - Continuous parameter ($\sigma > 0$)

μ - Continuous parameter ($\mu > 0$)

γ - Continuous location parameter

Domínio:

$\gamma < x < +\infty$

Anexo C10 – *Pearson* tipo 6

A distribuição do tipo *Pearson* tipo 6 apresenta a seguinte função densidade de probabilidade:

$$f(x) = \frac{\left((x-\gamma)/\beta\right)^{\alpha_1 - 1}}{\beta B(\alpha_1, \alpha_2)\left(1 + (x-\gamma)/\beta\right)^{\alpha_1 + \alpha_2}}$$

onde B representa a função Beta dada por:

$$B(\alpha_1, \alpha_2) = \int_0^1 t^{\alpha_1 - 1}(1-t)^{\alpha_2 - t}\, dt \qquad (\alpha_1, \alpha_2 > 0)$$

com parâmetros:

α_1 - Continuous shape parameter ($\alpha_1 > 0$)

α_2 - Continuous shape parameter ($\alpha_2 > 0$)

β - Continuous scale parameter ($\beta > 0$)

γ - Continuous location parameter

Dominio

$\gamma \leq x < +\infty$

Anexo C11 – *Pareto* tipo 2

A distribuição do tipo *Pareto* tipo 2 apresenta a seguinte função densidade de probabilidade:

$$f(x) = \frac{\alpha\beta^{\alpha}}{(x+\beta)^{\alpha+1}}$$

com parâmetros:

α - Continuous shape parameter ($\alpha > 0$)

β - Continuous scale parameter ($\beta > 0$)

Domínio:

$0 \le x < +\infty$

Anexo C12 – *Gamma*

A distribuição do tipo *Gamma* apresenta a seguinte função densidade de probabilidade:

$$f(x) = \frac{(x-\gamma)^{\alpha-1}}{\beta^{\alpha}\Gamma(\alpha)} \exp\left(-(x-\gamma)/\beta\right)$$

onde Γ representa a função *Gamma* dada por:

$$\Gamma(\alpha) = \int_0^{\infty} t^{\alpha-1} e^{-t}\, dt \qquad (\alpha > 0)$$

com parâmetros:

α - Continuous shape parameter ($\alpha > 0$)

β - Continuous scale parameter ($\beta > 0$)

γ - Continuous location parameter

Dominio

$\gamma \leq x < +\infty$